UNIVERSITÄT
DES
SAARLANDES

Fakultät für Mathematik und Informatik

Modulhandbuch

Cybersicherheit BSc

30. August 2020

Liste der Modulbereiche und Module

Modulbereich 1

Grundlagen der Mathematik

Mathematik für Informatiker 1 MfI1

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
1	**6**	**jedes Wintersemester**	**1 Semester**	**6**	**9**

Modulverantwortliche/r Prof. Dr. Joachim Weickert

Dozent/inn/en Prof. Dr. Joachim Weickert
Prof. Dr. Frank-Olaf Schreyer
Prof. Dr. Mark Groves

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen
- Teilnahme an den Übungen und Bearbeitung der wöchentlichen Übungsaufgaben (50 Prozent der Übungspunkte werden zur Klausurteilnahme benötigt)
- Bestehen der Abschlussklausur oder der Nachklausur

Lehrveranstaltungen / SWS
```
  4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS
```

Arbeitsaufwand
```
   90 h Präsenzstudium
+ 180 h Eigenstudum
= 270 h (= 9 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch

Lernziele / Kompetenzen

- Erarbeitung von mathematischem Grundlagenwissen, das im Rahmen eines Informatik- bzw. Bioinformatikstudiums benötigt wird
- Fähigkeit zur Formalisierung und Abstraktion
- Befähigung zur Aneignung weiteren mathematischen Wissens mit Hilfe von Lehrbüchern

Inhalt

Die Zahlen geben die Gesamtzahl der Doppelstunden an.

```
DISKRETE MATHEMATIK UND EINDIMENSIONALE ANALYSIS

A. Grundlagen der diskreten Mathematik (8)
    1. Mengen (1)
    2. Logik (1)
    3. Beweisprinzipien, incl. vollst. Induktion (1)
    4. Relationen (1)
    5. Abbildungen (2)
        - injektiv, surjektiv, bijektiv
        - Mächtigkeit, Abzählbarkeit
        - Schubfachprinzip
    6. Primzahlen und Teiler (1)
    7. Modulare Arithmetik (1

B. Eindimensionale Analysis (22)
```

B.1 Zahlen, Folgen und Reihen (8)
 8. Axiomatik der reellen Zahlen, sup, inf (1)
 9. Komplexe Zahlen (1)
10. Folgen (1 1/2)
11. Landau'sche Symbole (1/2)
12. Reihen: Konvergenzkriterien, absolute Kgz. (2)
13. Potenzreihen (1/2)
14. Zahlendarstellungen (1/2)
15. Binomialköffizienten und Binomialreihe (1)

B.2 Eindimensionale Differentialrechnung (8)
16. Stetigkeit (1)
17. Elementare Funktionen (1)
18. Differenzierbarkeit (1 1/2)
19. Mittelwertsätze und L'Hospital (1/2)
20. Satz von Taylor (1)
21. Lokale Extrema, Konvexität, Kurvendiskussion (2)
22. Numerische Differentiation (1)

B.3 Eindimensionale Integralrechnung (6)
23. Das bestimmte Integral (2)
24. Das unbestimmte Integral und die Stammfunktion (1)
25. Uneigentliche Integrale (1)
26. Numerische Verfahren zur Integration (1)
27. Kurven und Bogenlänge (1)

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet

Mathematik für Informatiker 2 — MfI2

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2	6	jedes Sommersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Joachim Weickert

Dozent/inn/en Prof. Dr. Joachim Weickert
Prof. Dr. Frank-Olaf Schreyer
Prof. Dr. Mark Groves

Zulassungsvoraussetzungen Mathematik für Informatiker 1 (empfohlen)

Leistungskontrollen / Prüfungen
- Teilnahme an den Übungen und Bearbeitung der wöchentlichen Übungsaufgaben (50 Prozent der Übungspunkte werden zur Klausurteilnahme benötigt)
- Bestehen der Abschlussklausur oder der Nachklausur

Lehrveranstaltungen / SWS
```
  4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS
```

Arbeitsaufwand
```
  90 h Präsenzstudium
+ 180 h Eigenstudum
= 270 h (= 9 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch

Lernziele / Kompetenzen

- Erarbeitung von mathematischem Grundlagenwissen, das im Rahmen eines Informatik- bzw. Bioinformatikstudiums benötigt wird
- Fähigkeit zur Formalisierung und Abstraktion
- Befähigung zur Aneignung weiteren mathematischen Wissens mit Hilfe von Lehrbüchern

Inhalt

```
C. Algebraische Strukturen (5)
    29. Gruppen (2)
    30. Ringe und Körper (1)
    31. Polynomringe über allgemeinen Körpern (1/2)
    32. Boole'sche Algebren (1/2)

D. Lineare Algebra (21)
    33. Vektorräume (2)
        - Def., Bsp.,
        - lineare Abb.
        - Unterraum,
        - Erzeugnis, lineare Abhängigkeit, Basis, Austauschsatz
    34. Lineare Abb. (Bild, Kern) (1)
    35. Matrixschreibweise für lineare Abbildungen (1 1/2)
        - Interpretation als lineare Abbildungen
        - Multiplikation durch Hintereinanderausführung
        - Ringstruktur
```

    - Inverses
36. Rang einer Matrix (1/2)
37. Gauss-Algorithmus für lineare Gleichungssysteme: (2)
    - Gausselimination (1)
    - Lösungstheorie (1)
38. Iterative Verfahren für lineare Gleichungssysteme (1)
39. Determinanten (1)
40. Euklidische Vektorräume, Skalarprodukt (1)
41. Funktionalanalytische Verallgemeinerungen (1)
42. Orthogonalität (2)
43. Fourierreihen (1)
44. Orthogonale Matrizen (1)
45. Eigenwerte und Eigenvektoren (1)
46. Eigenwerte und Eigenvektoren symmetrischer Matrizen (1)
47. Quadratische Formen und positiv definite Matrizen (1)
48. Quadriken (1)
50. Matrixnormen und Eigenwertabschätzungen (1)
51. Numerische Berechnung von Eigenwerten und Eigenvektoren (1)

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet

Modulbereich 2

Grundlagen der Informatik

Big Data Engineering BDE

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Sommersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Jens Dittrich

Dozent/inn/en Prof. Dr. Jens Dittrich

Zulassungsvoraussetzungen keine (siehe aber „Empfohlene Vorkenntnisse“ weiter unten)

Leistungskontrollen / Prüfungen Erfolgreiche Teilnahme an den Übungen/Projekt berechtigt zur Teilnahme an der Abschlussklausur (bzw. Studienarbeit).

Lehrveranstaltungen / SWS

```
  2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS
```

Arbeitsaufwand

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

Modulnote Wird aus Leistungen in Klausuren (alternativ Studienarbeit), Übungen, ggf. Projekt ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekanntgegeben.

Sprache Deutsch

Lernziele / Kompetenzen

Die Vorlesung vermittelt grundlegende Kenntnisse über fundamental Konzepte von Datenmanagement und Datenanalyse im Kontext von Big Data und Data Science.

Im Rahmen der Übungen kann während des Semesters ein durchgehendes Projekt durchgeführt. Dies kann zum Beispiel ein soziales Netzwerk (im Stil von Facebook) sein bzw. jedes andere Projekt, in dem Techniken des Datenmanagements eingeübt werden können (z.B. naturwissenschaftliche Daten, Bilddaten, andere Webapplikationen, etc.). Zunächst wird dieses Projekt in E/R modelliert, dann umgesetzt und implementiert in einem Datenbankschema. Danach wird das Projekt erweitert, um auch unstrukturierte Daten verwalten und analysieren zu können. Insgesamt werden so an einem einzigen Projekt alle fundamentalen Techniken gezeigt, die für das Verwalten und Analysieren von Daten wichtig sind.

Inhalt

```
1 Einführung und Einordnung
    Einordnung und Abgrenzung: Data Science
    Wert von Daten: Das Gold des 21. Jahrhunderts
    Bedeutung von Datenbanksystemen
    Architekturen: 2-Tier, 3-Tier, etc
    Was sind eigentlich Daten?
    Modellierung vs Realität
    Kosten mangelhafter Modellierung
    Datenbanksystem nutzen vs selbst entwickeln
    Positive Beispiele für Apps
    Anforderungen
    Literaturhinweise
    Vorlesungsmodus

2  Datenmodellierung
```

Motivation
E/R
Relationales Modell
Hierarchische Daten
Graphen und RDF
Redundanz, Normalisierung, Denormalisierung
Objektrelationale DBMS

3 Anfragesprachen
Relationale Algebra
Hierarchische Anfragesprachen
Graphorientierte Anfragesprachen

4 SQL
Grundlagen
Zusammenhang mit relationaler Algebra
PostgreSQL
Integritätsbedingungen
Transaktionskonzept
ACID
Sichten (und access control lists)

5 Implementierungstechniken
Übersicht
vom WAS zum WIE
Kosten verschiedener Operationen
EXPLAIN
Physisches Design
Indexe, Tuning
Datenbank-Tuning
Regelbasierte Anfrageoptimierung
Kostenbasierte Anfrageoptimierung
Machine Learning als Anfrageoptimierungstechnik

6 Zeitliche und räumliche Daten
als Teil des Schemas
as of/time travel
append-only und Streaming
Versioning
Snapshotting (Software und OS-basiert)
Differential Files/LSM et al
Publish/Subscribe
Indexstrukturen

7 Recovery, Durability, Archivierung
Grundproblematik
Vergessen vs Komprimieren vs Kondensieren
Heiße vs kalte Daten
Archivierung
Redundanz
Implementierungsaspekte
UNDO/REDO
Logging

8 Nebenläufigkeitskontrolle
Serialisierbarkeitstheorie
Isolationslevels
Verteilte Datenbanksysteme: Sharding, HP, VP, permissioned Blockchains
Implementierungsaspekte

9 ETL und Data Cleaning

```
    Datenbankschnittstellen: JDBC et al
    Textdatenbanken: CSV, SQlite
    Data Warehousing
    Schema Matching
    Reporting
    Data Cleaning
    Denormalisierung, Caching, Materialisierung
    Workflows
    ETL und Data Science in Data Science und Machine Learning

10 Big Data
    Was ist eigentlich Big Data?
    Big Data vs Privatheit
    Beispiele: Zusammenführen von Daten
    Physische Barrieren

11 NoSQL
    Key/Value Stores
    KeyDocument Stores: MongoDB
    MapReduce
    Flink
    Spark
```

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Weitere Informationen

Dieses Modul wurde früher auch unter dem Namen *Informationssysteme* geführt.

Empfohlene Vorkenntnisse: Programmierung 1, Programmierung 2, Softwarepraktikum oder Projektpraktikum, Mathematik für Informatiker 1, sowie Grundzüge von Algorithmen und Datenstrukturen.

Elements of Machine Learning — ElML

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	jedes Wintersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Jilles Vreeken
Prof. Dr. Isabel Valera

Dozent/inn/en Prof. Dr. Bernt Schiele
Prof. Dr. Jilles Vreeken
Prof. Dr. Isabel Valera

Zulassungsvoraussetzungen Die Vorlesung setzt grundlegende Kenntnisse in Statistik und linearer Algebra voraus. Ist deshalb ratsam, MfI2 und das Statistics Lab erfolgreich abgeschlossen zu haben. Die Übungen verwenden die Programmiersprache R und grundlegende Kenntnisse sind hilfreich. Zur Vorbereitung sind die folgenden Materialen nützlich: „R for Beginners“ von Emmanuel Paradis (insbesondere Kapitel 1, 2, 3 und 6) und „An introduction to R“ (Venables/Smith).

Leistungskontrollen / Prüfungen Voraussetzung zur Zulassung zur Prüfung sind 50% der Punkte der theoretischen und praktischen Aufgaben auf den Übungsblättern. Die Prüfungen finden, je nach Teilnehmerzahl, schriftlich oder mündlich statt. Die genauen Modalitäten werden in den ersten zwei Wochen der Vorlesung bekannt gegeben.

Lehrveranstaltungen / SWS

```
  2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS
```

Arbeitsaufwand

```
   60 h Präsenzstudium
+ 120 h Eigenstudum
= 180 h (= 6 ECTS)
```

Modulnote Siehe ”Leistungskontrollen/Prüfungen”

Sprache English

Lernziele / Kompetenzen

In diesem Kurs werden grundlegende Konzepte des maschinellen Lernens behandelt, wobei der Schwerpunkt auf statistischen Methoden liegt. Der Kurs vermittelt die nötigen Fähigkeiten um für einen gegebenen Datensatz geeignete statistische Methoden für dessen Analyse auszuwählen, anzuwenden, und die Qualität der Resultate zu bewerten. Der Kurs behandelt sowohl sowohl theoretische als auch praktische Aspekte des maschinellen Lernens, legt den Fokus jedoch auf praktische Aspekte.

Die Vorlesung folgt im Großen und Ganzen dem Buch “An Introduction to Statistical Learning with Applications in R (2013)”. In einigen Fällen erhält der Kurs zusätzliches Material aus dem Buch The Elements of Statistical Learning, Springer (second edition, 2009). Das erste Buch ist der einleitende Text, das zweite behandelt fortgeschrittenere Themen. Beide Bücher sind als kostenlose PDFs erhältlich. Es wird durchschnittlich eine Vorlesung pro Woche (90 Minuten) und alle zwei Wochen (90 Minuten) ein Tutorium angeboten.

Inhalt

Die Vorlesung behandelt grundlegende Methoden des maschinellen Lernens, insbesondere folgende Inhalte:

- Introduction to statistical learning
- Overview over Supervised Learning
- Linear Regression

- Linear Classification
- Splines
- Model selection and estimation of the test errors
- Maximum-Likelihood Methods
- Additive Models
- Decision trees
- Boosting
- Dimensionality reduction
- Unsupervised learning

Literaturhinweise

Grundzüge der Theoretischen Informatik TI

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	jedes Wintersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Raimund Seidel

Dozent/inn/en Prof. Dr. Raimund Seidel
Prof. Dr. Bernd Finkbeiner
Prof. Dr. Kurt Mehlhorn
Prof. Dr. Markus Bläser

Zulassungsvoraussetzungen *Programmierung 1* und *2* und *Mathematik für Informatiker 1* und *2* oder vergleichbare Veranstaltungen der Mathematik (empfohlen)

Leistungskontrollen / Prüfungen Erfolgreiche Bearbeitung der Übungsaufgaben berechtigt zur Klausurteilnahme.

Lehrveranstaltungen / SWS
```
  4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS
```

Arbeitsaufwand
```
   90 h Präsenzstudium
+ 180 h Eigenstudum
= 270 h (= 9 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch

Lernziele / Kompetenzen

Die Studierenden kennen verschiedene Rechenmodelle und ihre relativen Stärken und Mächtigkeiten.

Sie können für ausgewählte Probleme zeigen, ob diese in bestimmten Rechenmodellen lösbar sind oder nicht.

Sie verstehen den formalen Begriff der Berechenbarkeit wie auch der Nicht-Berechenbarkeit.

Sie können Probleme aufeinander reduzieren.

Sie sind vertraut mit den Grundzügen der Ressourcenbeschränkung (Zeit, Platz) für Berechnungen und der sich daraus ergebenden Komplexitätstheorie.

Inhalt

Die Sprachen der Chomsky Hierarchie und ihre verschiedenen Definitionen über Grammatiken und Automaten; Abschlusseigenschaften; Klassifikation von bestimmten Sprachen („Pumping lemmas“);

Determinismus und Nicht-Determinismus;

Turing Maschinen und äquivalente Modelle von allgemeiner Berechenbarkeit (z.B. µ-rekursive Funktionen, Random Access Machines) Reduzierbarkeit, Entscheidbarkeit, Nicht-Entscheidbarkeit;

Die Komplexitätsmaße Zeit und Platz; die Komplexitätsklassen P und NP;

Grundzüge der Theorie der NP-Vollständigkeit

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Grundzüge von Algorithmen und Datenstrukturen — GrAlgoDat

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	jedes Wintersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Raimund Seidel

Dozent/inn/en Prof. Dr. Raimund Seidel
Prof. Dr. Kurt Mehlhorn
Prof. Dr. Markus Bläser

Zulassungsvoraussetzungen *Programmierung 1* und *2*, und *Mathematik für Informatiker 1* und *2* oder vergleichbare Veranstaltungen der Mathematik (empfohlen)

Leistungskontrollen / Prüfungen Erfolgreiche Bearbeitung der Übungsblätter berechtigt zur Klausurteilnahme.

Lehrveranstaltungen / SWS

```
  2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS
```

Arbeitsaufwand

```
   60 h Präsenzstudium
+ 120 h Eigenstudum
= 180 h (= 6 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch

Lernziele / Kompetenzen

Die Studierenden lernen die wichtigsten Methoden des Entwurfs von Algorithmen und Datenstrukturen kennen: Teile-und-Herrsche, Dynamische Programmierung, inkrementelle Konstruktion, „Greedy“, Dezimierung, Hierarchisierung, Randomisierung. Sie lernen Algorithmen und Datenstrukturen bzgl. Zeit- und Platzverbrauch für das übliche RAM Maschinenmodell zu analysieren und auf Basis dieser Analysen zu vergleichen. Sie lernen verschiedene Arten der Analyse (schlechtester Fall, amortisiert, erwartet) einzusetzen.

Die Studierenden lernen wichtige effiziente Datenstrukturen und Algorithmen kennen. Sie sollen die Fähigkeit erwerben, vorhandene Methoden durch theoretische Analysen und Abwägungen für ihre Verwendbarkeit in tatsächlich auftretenden Szenarien zu prüfen. Ferner sollen die Studierenden die Fähigkeit trainieren, Algorithmen und Datenstrukturen unter dem Aspekt von Performanzgarantien zu entwickeln oder anzupassen

Inhalt

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Nebenläufige Programmierung NP

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	**6**	**jedes Sommersemester**	**1 Semester**	**4**	**6**

Modulverantwortliche/r Prof. Dr.-Ing. Holger Hermanns

Dozent/inn/en Prof. Dr.-Ing. Holger Hermanns
Prof. Bernd Finkbeiner, Ph.D
Prof. Dr. Verena Wolf

Zulassungsvoraussetzungen *Programmierung 1* und *2*, *Softwarepraktikum*, und *Grundzüge der Theoretischen Informatik* (empfohlen)

Leistungskontrollen / Prüfungen Zwei Klausuren (Mitte und Ende der Vorlesungszeit), praktisches Projekt.

Nachklausuren finden innerhalb der letzten Wochen
vor Vorlesungsbeginn des Folgesemesters statt..

Lehrveranstaltungen / SWS **Element T – Theorie (2 SWS):**
8 Vorlesungen: 6 Wochen
4 Übungen: 6 Wochen

Element A – Anwendung (2 SWS):
9 Vorlesungen: 6 Wochen
4 Übungen: 6 Wochen

Element P – Praxis (Eigenstudium):
Semesterbegleitend 8 schriftliche Reflektionen (Prüfungsvorleistungen), anschließend Projektarbeit über ca. 2 Wochen

= 4 SWS

Arbeitsaufwand **Element T:**
24 h Präsenz, 36h Selbststudium

Element A:
26 h Präsenz, 34h Selbststudium

Element P:
60 h Selbststudium

```
   50 h Präsenzstudium
+ 130 h Eigenstudum
= 180 h (= 6 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren (im Anschluss an die Elemente T und A), sowie den Prüfungsvorleistungen (Element P) ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben. Alle Modulelemente sind innerhalb eines Prüfungszeitraumes erfolgreich zu absolvieren.

Sprache Deutsch

Lernziele / Kompetenzen

Die Teilnehmer lernen die Nebenläufigkeit von Prozessen als ein weitreichendes, grundlegendes Prinzip in der Theorie und Anwendung der modernen Informatik kennen. Durch die Untersuchung und Verwendung unterschiedlicher formaler Modelle gewinnen die Teilnehmer ein vertieftes Verständnis von Nebenläufigkeit. Außerdem lernen die Teilnehmer wichtige formale Konzepte der Informatik korrekt anzuwenden. Das im ersten Teil der Veranstaltung erworbene theoretische Wissen wird in der zweiten Hälfte in der (Programmier-) Praxis angewendet. Dabei lernen die Teilnehmer verschiedene Phänomene des nebenläufigen Programmierens in den formalen Modellen zu beschreiben und mit deren Hilfe konkrete Lösungen für die Praxis abzuleiten. Des Weiteren werden die Teilnehmer in der Praxis existierende Konzepte auf diese Art auf ihre Verlässlichkeit hin untersuchen.

Inhalt

Nebenläufigkeit als Konzept

- Potentieller Parallelismus
- Tatsächlicher Parallelismus
- Konzeptioneller Parallelismus

Nebenläufigkeit in der Praxis

- Objektorientierung
- Betriebssysteme
- Multi-core Prozessoren, Coprozessoren
- Programmierte Parallelität
- Verteilte Systeme (Client-Server, Peer-to-Peer, Datenbanken, Internet)

Die Schwierigkeit von Nebenläufigkeit

- Ressourcenkonflikte
- Fairness
- Gegenseitiger Auschluss
- Verklemmung (Deadlock)
- gegenseitige Blockaden (Livelock)
- Verhungern (Starvation)

Grundlagen der Nebenläufigkeit

- Sequentielle Prozesse
- Zustände, Ereignisse und Transitionen
- Transitionssysteme
- Beobachtbares Verhalten
- Determinismus vs. Nicht-Determinismus – Algebren und Operatoren

CCS: Der Kalkül kommunizierender Prozesse

- Konstruktion von Prozessen: Sequenz, Auswahl, Rekursion
- Nebenläufigkeit
- Interaktion
- Strukturelle operationelle Semantik
- Gleichheit von Beobachtungen
- Implementierungsrelationen
- CCS mit Datentransfer

Programmieren von Nebenläufigkeit

- pseuCo
- Message-Passing in pseuCo und Go
- Shared-memory in pseuCo und Java
- Shared Objects und Threads in Java
- Shared Objects und Threads als TransitionssystemeFakultät für Mathematik und Informatik

Analyse und Programmierunterstützung

- Erkennung von Verklemmungen
- Zusicherung von Sicherheit und Lebendigkeit
- Model-Basiertes Design von Nebenläufigkeit
- Software Architekturen für Nebenläufigkeit

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Programmierung 1 — Prog1

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
1	6	jedes Wintersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Gert Smolka

Dozent/inn/en Prof. Dr. Gert Smolka
Prof. Dr.-Ing. Holger Hermanns
Prof. Bernd Finkbeiner, Ph.D

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen
- zwei Klausuren (Mitte und Ende der Vorlesungszeit)
- Die Note wird aus den Klausuren gemittelt und kann durch Leistungen in den Übungen verbessert werden.
- Eine Nachklausur findet innerhalb der letzten beiden Wochen vor Vorlesungsbeginn des Folgesemesters statt.

Lehrveranstaltungen / SWS
```
  4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS
```

Arbeitsaufwand
```
  90 h Präsenzstudium
+ 180 h Eigenstudum
= 270 h (= 9 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch

Lernziele / Kompetenzen

- höherstufige, getypte funktionale Programmierung anwenden können
- Verständnis rekursiver Datenstrukturen und Algorithmen, Zusammenhänge mit Mengenlehre
- Korrektheit beweisen und Laufzeit abschätzen
- Typabstraktion und Modularisierung verstehen
- Struktur von Programmiersprachen verstehen
- einfache Programmiersprachen formal beschreiben können
- einfache Programmiersprachen implementieren können
- anwendungsnahe Rechenmodelle mit maschinennahen Rechenmodellen realisieren können
- Praktische Programmiererfahrung, Routine im Umgang mit Interpretern und Übersetzern

Inhalt

- Funktionale Programmierung
- Algorithmen und Datenstrukturen (Listen, Bäume, Graphen; Korrektheitsbeweise; asymptotische Laufzeit)
- Typabstraktion und Module
- Programmieren mit Ausnahmen
- Datenstrukturen mit Zustand
- Struktur von Programmiersprachen (konkrete und abstrakte Syntax, statische und dynamische Syntax)
- Realisierung von Programmiersprachen (Interpreter, virtuelle Maschinen, Übersetzer)

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet

Programmierung 2 — Prog2

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2	**6**	**jedes Sommersemester**	**1 Semester**	**6**	**9**

Modulverantwortliche/r Prof. Dr. Sebastian Hack

Dozent/inn/en Prof. Dr. Sebastian Hack
Prof. Dr. Jörg Hoffmann

Zulassungsvoraussetzungen *Programmierung 1* und *Mathematik für Informatiker 1* und Mathematikveranstaltungen im Studiensemester oder vergleichbare Kenntnisse aus sonstigen Mathematikveranstaltungen (empfohlen)

Leistungskontrollen / Prüfungen Prüfungsleistungen werden in zwei Teilen erbracht, die zu gleichen Teilen in die Endnote eingehen. Um die Gesamtveranstaltung zu bestehen, muss jeder Teil einzeln bestanden werden.

Im **Praktikumsteil** müssen die Studierenden eine Reihe von Programmieraufgaben selbstständig implementieren. Diese Programmieraufgaben ermöglichen das Einüben der Sprachkonzepte und führen außerdem komplexere Algorithmen und Datenstrukturen ein. Automatische Tests prüfen die Qualität der Implementierungen. Die Note des Praktikumsteils wird maßgeblich durch die Testergebnisse bestimmt.

Im **Vorlesungsteil** müssen die Studierenden Klausuren absolvieren und Übungsaufgaben bearbeiten. Die Aufgaben vertiefen dabei den Stoff der Vorlesung. Die Zulassung zu der Klausur hängt von der erfolgreichen Bearbeitung der Übungsaufgaben ab.

Im Praktikumsteil kann eine Nachaufgabe angeboten werden

Lehrveranstaltungen / SWS

```
  4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS
```

Arbeitsaufwand

```
   90 h Präsenzstudium
+ 180 h Eigenstudum
= 270 h (= 9 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben

Sprache Deutsch

Lernziele / Kompetenzen

Die Studierenden lernen die Grundprinzipien der imperativen /objektorientierten Programmierung kennen. Dabei wird primär Java als Programmiersprache verwendet.

In dieser Vorlesung lernen sie:

- wie Rechner Programme ausführen
- Die Grundlagen imperativer und objektorientierter Sprachen
- kleinere, wohlstrukturierte Programme in C zu schreiben
- mittelgroße objektorientierte Systeme in Java zu implementieren und zu testen
- sich in wenigen Tagen eine neue imperative/objektorientierte Sprache anzueignen, um sich in ein bestehendes Projekt einzuarbeiten Inhalt

Inhalt

- Imperatives Programmieren
- Objekte und Klassen
- Klassendefinitionen
- Objektinteraktion
- Objektsammlungen
- Objekte nutzen und testen
- Vererbung
- Dynamische Bindung
- Fehlerbehandlung
- Klassendesign und Modularität
- Systemnahe Programmierung

sowie spezifische Vorlesungen für die Programmieraufgaben.

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet

Statistics Lab

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2	6	jedes Sommersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Verena Wolf
Prof. Dr. Vera Demberg

Dozent/inn/en Prof. Dr. Verena Wolf
Prof. Dr. Vera Demberg

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen mündliche oder schriftliche Prüfung

Lehrveranstaltungen / SWS
```
  2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS
```

Arbeitsaufwand
```
   60 h Präsenzstudium
+ 120 h Eigenstudum
= 180 h (= 6 ECTS)
```

Modulnote Wird aus Leistungen in der Klausur, sowie den Prüfungsvorleistungen ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben. Alle Modulelemente sind innerhalb eines Prüfungszeitraumes erfolgreich zu absolvieren.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

- Verständnis der mathematischen Konzepte von Zufallsvariablen und Verteilungen
- Verständnis und Anwendung von Methoden der Punkt-und Intervalschätzung, statistischer Tests
- Verständnis der mathematischen Konzepte von zustandsdiskreten Markovprozessen und Verwendung solcher Prozesse zur Beschreibung von realen Phänomenen

Inhalt

Probabilities and Discrete Random Variables

- Probability
- discrete RVs
- expectation, variance and quantiles (also visualization of them)
- higher moments
- important discrete probability distributions
- Generating discrete random variates Continuous Random Variables and Laws of Large Numbers
- σ-algebras (very lightweight)
- Continuous Random Variables
- Important Continuous Distributions
- generating continuous random variates
- Chebyshev's inequality
- Weak/Strong Law of Large Numbers
- Central Limit Theorem

Multidimensional Probability Distributions

- joint probability distribution

- conditional probability distribution
- Bayes' Theorem
- covariance and correlation
- independence
- important multidimensional probability distributions

Point Estimation

- (generalized) method of moments
- maximum likelihood estimation
- Bayesian inference (posterior mean/median, MAP)
- Kernel density estimation
- OLS estimator (this is simple regression but should be mentioned here!)
- (shortly: model selection)

Interval Estimation

- confidence intervals for sample mean/variance
- confidence intervals for MLE
- bootstrap confidence interval
- Bayesian credible interval

Statistical Testing

- Level α tests (Z-Test, T-Test)
- p-value
- chi-squared tests, Fisher test
- multiple testing (Bonferroni correction, Holm-Bonferroni method, Benjamini-Hochberg, etc)

Discrete-time Markov chains (only if time)

- transient distributions
- equilibrium distributions
- Monte-Carlo simulation

HMMs

- Baum-Welch-Algorithmus
- Viterbi-Algorithmus

Literaturhinweise

Systemarchitektur SysArch

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Sommersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Jan Reineke

Dozent/inn/en Prof. Dr. Jan Reineke

Zulassungsvoraussetzungen *Programmierung 1* und *Mathematik für Informatiker 1* oder vergleichbare Veranstaltungen der Mathematik (empfohlen)

Leistungskontrollen / Prüfungen **Studienleistungen:** die Vorlesungen hören, nach bearbeiten und gegebenenfalls verstehen; die Übungen allein oder in Gruppen bearbeiten; erfolgreich bearbeitete Übungen in der Übungsgruppe vortragen.

Prüfungsleistungen: erfolgreiche Bearbeitung von 50 % der Übungsaufgaben berechtigt zur Teilnahme an den Klausuren. Bestehen von zwei aus drei Klausuren.

Lehrveranstaltungen / SWS

```
  4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS
```

Arbeitsaufwand

```
   90 h Präsenzstudium
+ 180 h Eigenstudum
= 270 h (= 9 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch

Lernziele / Kompetenzen

Die Studierenden sollen die Funktionsweise, die Eigenschaften und die Entwurfsprinzipien von Rechnerarchitekturen und Betriebssystemen kennen lernen.

Inhalt

1. Hardware
    a. Boole'sche Algebra und Schaltkreise
    b. Elementare Rechnerarithmetik
    c. ALU (Konstruktion und Korrektheit)
    d. Sequentieller vereinfachter DLX-Prozessor (Konstruktion und Korrektheit)

2. Betriebssystemkern
    a. Virtualisierung
    b. Ressourcen-Verwaltung, Speicher, Prozessor
    c. Scheduling
    d. Datei-System

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Modulbereich 3

Praktika

Softwarepraktikum SoPra

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2-3	**6**	**Vorlesungsfr. Zeit nach dem SS**	**7 Wochen**	**BLOCK**	**9**

Modulverantwortliche/r Prof. Dr. Sven Apel

Dozent/inn/en Prof. Dr. Sven Apel

Zulassungsvoraussetzungen Die Teilnahme am Softwarepraktikum setzt umfangreiche Programmierkenntnisse voraus, wie sie in den Vorlesungen *Programmierung 1* und *Programmierung 2* vermittelt werden.

Für die Teilnahme am Softwarepraktikum werden eigene Laptops benötigt, die selbst mitgebracht werden müssen.

Leistungskontrollen / Prüfungen Das Ziel des Softwarepraktikums ist es, in einer Gruppe von Studierenden ein nichttriviales Softwaresystem zu erstellen. Dazu müssen eine Reihe von Dokumenten (Entwurf, Quellcode, Tests, etc.) erstellt und abgegeben werden. Bewertet wird die Korrektheit und Qualität der Dokumente sowie die fristgerechte Abgabe.

Das Softwarepraktikum gliedert sich in eine Übungsphase, eine Gruppenphase und eine Einzelphase. In der Übungsphase werden täglich Minitests zu den aktuellen Vorlesungsinhalten durchgeführt und bewertet.

In der Gruppenphase wird ein substantielles Softwaresystem im Team geplant, entworfen, implementiert und getestet. Um zur Gruppenphase zugelassen zu werden, müssen die Studierenden die Minitests der Übungsphase bestehen.

In der Einzelphase wird ein kleineres Softwaresystem oder eine Erweiterung eines bestehenden Systems (z.B. aus der Gruppenphase) von den Studierenden jeweils allein entwickelt. Voraussetzung für die Einzelphase ist die erfolgreiche Absolvierung der Gruppenphase.

Die Softwaresysteme der Gruppen- und Einzelphase, sowie die zugehörigen Dokumente (Entwurf, Quellcode, Tests, etc.), werden auf Basis der Prinzipien und Qualitätsstandards der Vorlesung bewertet. Genauere Prüfungsmodalitäten werden zu Beginn des Softwarepraktikums in der Vorlesung bekannt gegeben.

Lehrveranstaltungen / SWS
```
täglich Projektarbeit mit Betreuung
teilweise Vorlesung
```

Arbeitsaufwand
```
  25 h Vorlesung
+ 245 h Projektarbeit
= 270 h (= 9 ECTS)
```

Modulnote unbenotet

Sprache Deutsch

Lernziele / Kompetenzen

Die Studierenden erwerben die Fähigkeit, im Team zu arbeiten und Probleme des Software Engineerings zu lösen.

Die Studierenden wissen, welche Probleme beim Durchführen eines Softwareprojekts auftreten können, und wie diese gelöst werden können.

Sie können eine komplexe Aufgabenstellung eigenständig in ein Softwareprodukt umsetzen, das den Anforderungen des Kunden entspricht. Hierfür wählen sie einen passenden Entwicklungsprozess, der Risiken früh erkannt und minimiert, und wenden diesen an.

Sie sind vertraut mit Grundzügen des Softwareentwurfs wie schwache Kopplung, hohe Kohäsion, Geheimnisprinzip sowie Entwurfs- und Architekturmustern und sind in der Lage, einen Entwurf anhand dieser Kriterien zu erstellen, zu beurteilen und zu verbessern.

Sie beherrschen Techniken der Qualitätssicherung wie Testen und Debugging und wenden diese an.

Inhalt

- Softwareentwurf
- Softwaretesten
- Teamarbeit
- Debugging

Literaturhinweise

- Software Engineering. I. Sommerville, Addison-Wesley, 2004.
- Software Engineering: A Practioner's Approach. R. Pressman, McGraw Hill Text, 2001.
- Using UML: Software Engineering with Objects and Components. P. Stevens, et al., Addison-Wesley, 1999.
- UML Distilled. M. Fowler, et al., Addison-Wesley, 2000.
- Objects, Components and Frameworks with UML, D. D'Souza, et al., Addison-Wesley, 1999.
- Designing Object-Oriented Software. R. Wirfs-Brock, et al., Prentice Hall, 1990.
- Design Patterns. Elements of Reusable Object-Oriented Software. E. Gamma, et al., Addison Wesley, 1995.
- Pattern-Oriented Software Architecture: A System of Patterns. F. Buschmann, et al., Wiley, 1996.
- Head First Design Patterns. E. Freeman, et al. O'Reilly, 2004.
- Software Architecture: Perspectives on an Emerging Discipline. M. Shaw, et al., Prentice-Hall, 1996.
- Refactoring: Improving the Design of Existing Code. M. Fowler, et al., Addison-Wesley, 1999.
- Software Testing and Analysis: Process, Principles and Techniques. M. Pezze, Wiley. 2007.

Modulbereich 4

Spezialisierter Bereich Cybersicherheit

Cryptography — Crypto

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Michael Backes

Dozent/inn/en Prof. Dr. Markus Bläser
Dr. Nico Döttling

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

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

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

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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

Literaturhinweise

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

Cyber Security Project

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5	6	every semester	1 Semester	6	9

Modulverantwortliche/r Dr. Sven Bugiel

Dozent/inn/en Dozent/inn/en der Fachrichtung

Zulassungsvoraussetzungen Grundlegende Kenntnisse im jeweiligen Teilbereich der Informatik.

Leistungskontrollen / Prüfungen Projektarbeit, Projektdokumentation, Projektpräsentation

Lehrveranstaltungen / SWS

```
  2 SWS Vorlesung
+ 4 SWS Praktikum
= 6 SWS
```

Arbeitsaufwand

```
   30 h Präsenzstudium
+ 240 h Projektarbeit
= 270 h (= 9 ECTS)
```

Modulnote Das Modul ist insgesamt bestanden, wenn die Prüfungsleistungbestanden wurde.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Die Studierenden erwerben die Fähigkeit, im Team zu arbeiten und Probleme der Cybersicherheit zu lösen.Die Studierenden wissen, welche sicherheitskritischen Probleme auftreten können, und wie man damit umgeht.Sie sind vertraut mit Grundzügen der Cybersicherheit wie den grundlegenden kryptographischen Primitiven, der Schutz der Privatsphäre und der Systemsicherheit, sie können Cyberangriffe erkennen und entsprechende Maßnahmen treffen.

Inhalt

Siehe Lernziele/Kompetenzen

Literaturhinweise

Die Literatur zum Modul kann englisch- und/oder deutschsprachig sein und wird zu Beginn der Veranstaltung bekanntgegeben.

Weitere Informationen

Die Unterrichtssprache ist deutsch oder englisch und wird zu Beginn der Veranstaltung bekannt gegeben.

Foundations of Cyber Security 1 — CySec1

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
1	6	every winter semester	1 semester	6	9

Modulverantwortliche/r Dr. Ben Stock

Dozent/inn/en Dr. Ben Stock

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen Erfolgreiche Bearbeitung der Übungsaufgaben berechtigen zur Klausurteilnahme.

Lehrveranstaltungen / SWS

```
  2 SWS Vorlesung
+ 2 SWS Übung
+ 2 SWS Projekt
= 6 SWS
```

Arbeitsaufwand

```
   60 h Präsenzstudium
+ 120 h Eigenstudum
+  90 h Projekt
= 270 h (= 9 ECTS)
```

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache English

Lernziele / Kompetenzen

Die Studierenden kennen die rechtlichen Grundlagen des Strafgesetzbuches in Bezug auf Computersicherheit. Daneben kennen sie Grunlagen der Kryptographie, Netzwerksicherheit sowie der Privatspähre. Besonderer Fokus liegt dabei auf Netzwerksicherheit, so dass Studenten für sichere Kommunikation relevante Protokolle kennen und einsetzen können.

Inhalt

Grundlagen des StGB bzgl. Computersicherheit
Grundlagen der symmetrische und asymmetrischer Kryptographie sowie deren Einsatzgebiete
Kenntnisse von Hash-Funktionen und deren Eigenschaften
Netzwerkgrundlagen aller Schichten (nach TCP/IP Modell)
Sicherheitsprotokolle auf den einzelnen Netzwerkschichten
Grundlagen der Privatsphäre und Anonymität
Grundlagen der Web-Sicherheit

Literaturhinweise

Die Literatur zum Modul ist englischsprachig und wird zu Beginn der Veranstaltung bekannt gegeben.

Weitere Informationen

Programmieraufgaben am Computer. Übungsaufgaben auf Papier und in Gruppen an der Tafel.

Foundations of Cyber Security 2 — CySec2

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2	6	every summer semester	1 semester	4	6

Modulverantwortliche/r Prof. Dr. Christian Rossow

Dozent/inn/en Prof. Dr. Christian Rossow

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen Written exam, and possibly mid-term exams and/or graded exercise sheets

Lehrveranstaltungen / SWS

```
  2 SWS lectures
+ 2 SWS tutorial
= 4 SWS
```

Arbeitsaufwand

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

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

Sprache Englisch

Lernziele / Kompetenzen

Students know the foundations of security in software, operating systems and IT systems in general.

Inhalt

- Basic Introduction to Operating Systems
- Foundations of System Security
- Foundations of Software Security
- Foundations of Attack Detection and Defense

Literaturhinweise

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

Modulbereich 5

Seminare

Proseminar

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	jedes Semester	1 Semester	2	5

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Professoren der Fachrichtung

Zulassungsvoraussetzungen Grundlegende Kenntnisse im jeweiligen Teilbereich des Studienganges.

Leistungskontrollen / Prüfungen
- Diskussion in der Gruppe
- thematischer Vortrag
- kurze schriftliche Ausarbeitung

Lehrveranstaltungen / SWS 2 SWS Proseminar

Arbeitsaufwand
```
  30 h Präsenzstudium
+ 120 h Eigenstudum
= 150 h (= 5 ECTS)
```

Modulnote Die Modalitäten der Notenvergabe werden vom jeweiligen verantwortlichen Hochschullehrer festgelegt.

Sprache Deutsch

Lernziele / Kompetenzen

Die Studierenden haben am Ende der Veranstaltung ein profundes Verständnis aktueller oder fundamentaler Aspekte eines spezifischen Teilbereiches des Studienganges erlangt.

Sie haben Kompetenz im Verstehen einfacher wissenschaftlicher Aufsätze und im Präsentieren von wissenschaftlichen Erkenntnissen erworben.

Inhalt

Unter Anleitung werden folgende Punkte praktisch geübt:

- Lesen und Verstehen wissenschaftlicher Aufsätze
- Diskutieren der Aufsätze in der Gruppe
- Analysieren, Zusammenfassen und Wiedergeben des spezifischen Themas
- Präsentationstechnik
- Spezifische Vertiefung in Bezug auf das individuelle Thema des Seminars.

Literaturhinweise

dem Thema entsprechend

Weitere Informationen

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

Seminar

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5	**6**	**jedes Semester**	**1 Semester**	**2**	**7**

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Professoren der Fachrichtung

Zulassungsvoraussetzungen Grundlegende Kenntnisse im jeweiligen Teilbereich der Studienganges.

Leistungskontrollen / Prüfungen
- Beiträge zur Diskussion
- Thematischer Vortrag
- Schriftliche Ausarbeitung
- Mündliche Abschlussprüfung über das gesamte Themengebiet

Lehrveranstaltungen / SWS `2 SWS Seminar`

Arbeitsaufwand
```
  30 h Präsenzstudium
+ 180 h Eigenstudum
= 210 h (= 7 ECTS)
```

Modulnote Die Modalitäten der Notenvergabe werden vom verantwortlichen Hochschullehrer festgelegt.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

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

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

Inhalt

Praktisches Einüben von

- reflektierender wissenschaftlicher Arbeit,
- Analyse und Bewertung wissenschaftlicher Aufsätze,
- Verfassen eigener wissenschaftlicher Zusammenfassungen
- Diskussion der Arbeiten in der Gruppe
- Erarbeiten gemeinsamer Standards für wissenschaftliches Arbeite
- Präsentationstechnik

Spezifische Vertiefung in Bezug auf das individuelle Thema des Seminars.

Der typische Ablauf eines Seminars ist wie folgt:

- Vorbereitende Gespräche zur Themenauswahl
- Regelmäßige Treffen mit Diskussion ausgewählter Beiträge
- Vortrag und Ausarbeitung zu einem der Beiträge
- Mündliche Prüfung über das erarbeitete Themengebiet

Literaturhinweise

dem Thema entsprechend

Weitere Informationen

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

Modulbereich 6

Vertiefungsvorlesungen der Cybersicherheit

Advanced Public Key Cryptography APC

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Dr. Nico Döttling

Dozent/inn/en Dr. Nico Döttling

Zulassungsvoraussetzungen Cryptography

Leistungskontrollen / Prüfungen Mündliche Prüfung oder Abschlussklausur

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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

Literaturhinweise

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

Automated Debugging

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	**6**	**at least every two years**	**1 semester**	**4**	**6**

Modulverantwortliche/r Prof. Dr. Andreas Zeller

Dozent/inn/en Prof. Dr. Andreas Zeller

Zulassungsvoraussetzungen *Programmierung 1*, *Programmierung 2* and *Softwarepraktikum*

Leistungskontrollen / Prüfungen Projects and mini-tests

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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

Literaturhinweise

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

Generating Software Tests GST

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Prof. Dr. Andreas Zeller

Dozent/inn/en Prof. Dr. Andreas Zeller

Zulassungsvoraussetzungen Programming 1, Programming 2, Softwarepraktikum

Leistungskontrollen / Prüfungen Projekte und Mini-Tests

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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

Literaturhinweise

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

Machine Learning in Cyber Security — MLCySec

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	**6**	**occasional**	**1 semester**	**4**	**6**

Modulverantwortliche/r Prof. Dr. Mario Fritz

Dozent/inn/en Prof. Dr. Mario Fritz

Zulassungsvoraussetzungen *Data Science/Statistics Course*

Leistungskontrollen / Prüfungen Übungen, Projekt und mündliche Prüfung

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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

Literaturhinweise

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

Malware Analysis and Intrusion Detection IDS

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Prof. Dr. Christian Rossow

Dozent/inn/en Prof. Dr. Christian Rossow

Zulassungsvoraussetzungen *Security* or *Foundations of Cyber Security I + II*

Leistungskontrollen / Prüfungen Projekt und Abschlussklausur

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

Students know the inner workings of malicious software, learn ways how to analyze an unknown x86/x64 program, and gain knowledge of intrusion and anomaly detection systems.

Inhalt

- Inner workings and types of malicious software
- Software reverse engineering basics
- x86/x64 assembly basics
- Practical static and dynamic program analysis techniques
- Malware analysis project(s)
- Host-based and network-based Intrusion Detection Systems
- Anomaly Detection systems

Literaturhinweise

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

Mobile Security MOS

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Dr. Sven Bugiel

Dozent/inn/en Dr. Sven Bugiel

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

Leistungskontrollen / Prüfungen Schriftliche Abschlussklausur

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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

Literaturhinweise

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

Physical-Layer Security PHYSEC

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Dr. Nils-Ole Tippenhauer

Dozent/inn/en Dr. Nils-Ole Tippenhauer

Zulassungsvoraussetzungen *Security* or *Foundations of Cyber Security I + II*

Leistungskontrollen / Prüfungen Übungen und schriftliche Abschlussklausur

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

- 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

Inhalt

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

Literaturhinweise

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

Weitere Informationen

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

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Dr. Yang Zhang

Dozent/inn/en Dr. Yang Zhang

Zulassungsvoraussetzungen Machine Learning

Leistungskontrollen / Prüfungen Projekt und Abschlussklausur

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

Students know the inner workings of malicious software, learn ways how to analyze an unknown x86/x64 program, and gain knowledge of intrusion and anomaly detection systems.

Inhalt

- Privacy Quantification
- Differential Privacy
- Machine Learning and Privacy

Literaturhinweise

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

Reactive Synthesis RS

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Dr. Swen Jacobs

Dozent/inn/en Dr. Swen Jacobs

Zulassungsvoraussetzungen *Grundzüge der Theoretischen Informatik*

Leistungskontrollen / Prüfungen Projekt und schriftliche Abschlussklausur

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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

Literaturhinweise

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

Secure Web Development — SecWebDev

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Dr. Nils-Ole Tippenhauer

Dozent/inn/en Dr. Nils-Ole Tippenhauer

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen Projekt und schriftliche Abschlussklausur

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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)

Literaturhinweise

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

Weitere Informationen

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

Web Security WMS

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	occasional	1 semester	4	6

Modulverantwortliche/r Dr. Ben Stock

Dozent/inn/en Dr. Ben Stock

Zulassungsvoraussetzungen *Security* or *Foundations of Cyber Security I + II*

Leistungskontrollen / Prüfungen Projekt und schriftliche Abschlussklausur

Lehrveranstaltungen / SWS

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

Arbeitsaufwand

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

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

Sprache English

Lernziele / Kompetenzen

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.

Inhalt

- 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)

Literaturhinweise

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

Modulbereich 7

Bachelor-Seminar und -Arbeit

Bachelor-Seminar

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
6	6	jedes Semester	variabel	2	9

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Professoren der Fachrichtung

Zulassungsvoraussetzungen Erwerb von mindestens 120 CP

Leistungskontrollen / Prüfungen

- Aufbereitung der relevanten wissenschaftlichen Literatur
- Schriftliche Ausarbeitung der Aufgabenstellung der Bachelorarbeit
- Vortrag über die geplante Aufgabenstellung mit anschliessender Diskussion
- Aktive Teilnahme an der Diskussion

Lehrveranstaltungen / SWS 2 SWS Seminar

Arbeitsaufwand

```
  30 h Präsenzstudium (Seminar)
+  20 h Betreuung durch den Lehrstuhl
+ 220 h Eigenstudium
= 270 h (= 9 ECTS)
```

Modulnote benotet

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Im Bachelorseminar erwirbt der Studierende unter Anleitung die Fähigkeit zum wissenschaftlichen Arbeiten im Kontext eines angemessen Themengebietes.

Am Ende des Bachelorseminars sind die Grundlagen für eine erfolgreiche Anfertigung der Bachelorarbeit gelegt, und wesentliche Lösungsansätze bereits eruiert.

Das Bachelorseminar bereitet somit die Themenstellung und Ausführung der Bachelorarbeit vor.

Es vermittelt darüber hinaus praktische Fähigkeiten des wissenschaftlichen Diskurses. Diese Fähigkeiten werden durch die aktive Teilnahme an einem Lesekreis vermittelt, in welchem die Auseinandersetzung mit wissenschaftlich anspruchsvollen Themen geübt wird.

Inhalt

Auf der Grundlage des ”state-of-the-art” werden die Methoden der Informatik systematisch unter Anleitung angewendet.

Literaturhinweise

Dem Themengebiet entsprechende wissenschaftliche Artikel in enger Absprache mit dem Dozenten

Bachelor-Arbeit

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
6	6	jedes Semester	3 Monate	-	12

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Professoren der Fachrichtung

Zulassungsvoraussetzungen Erfolgreicher Abschluss des *Bachelor-Seminars*

Leistungskontrollen / Prüfungen Schriftliche Ausarbeitung. Sie beschreibt sowohl das Ergebnis der Arbeit als auch den Weg, der zu dem Ergebnis führte. Der eigene Anteil an den Ergebnissen muss klar erkennbar sein. Außerdem Präsentation der Bachelorarbeit in einem Kolloquium, in dem auch die Eigenständigkeit der Leistung des Studierenden überprüft wird.

Lehrveranstaltungen / SWS keine

Arbeitsaufwand

```
  20 h Betreung durch den Lehrstuhl
+ 340 h Eigenstudium
= 360 h (= 12 ECTS)
```

Modulnote Beurteilung der Bachelorarbeit

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Die Bachelor-Arbeit ist eine Projektarbeit, die unter Anleitung ausgeführt wird. Sie zeigt, dass der Kandidat/die Kandidatin in der Lage ist, innerhalb einer vorgegebenen Frist ein Problem aus dem Gebiet der Informatik unter Anleitung zu lösen und die Ergebnisse zu dokumentieren.

Inhalt

Auf der Grundlage des ”state-of-the-art” wird die systematische Anwendung der Methoden der Informatik dokumentiert.

Literaturhinweise

Je nach Thema in Absprache mit dem betreuenden Hochschullehrer