Sicherheitsgarantien

zuverlässig nachweisen

Die Sicherheit von Softwaresystemen zuverlässig zu garantieren stellt eine große Herausforderung dar, die sowohl theoretische als auch praktische Aspekte beinhaltet. Im Fachgebiet Modellierung und Analyse von Informationssystemen (MAIS) werden mathematisch fundierte Methoden und Werkzeuge entwickelt, die eine präzise Modellierung und einen verlässlichen Nachweis von Sicherheitsanforderungen ermöglichen.

Reliable assurance of security guarantees

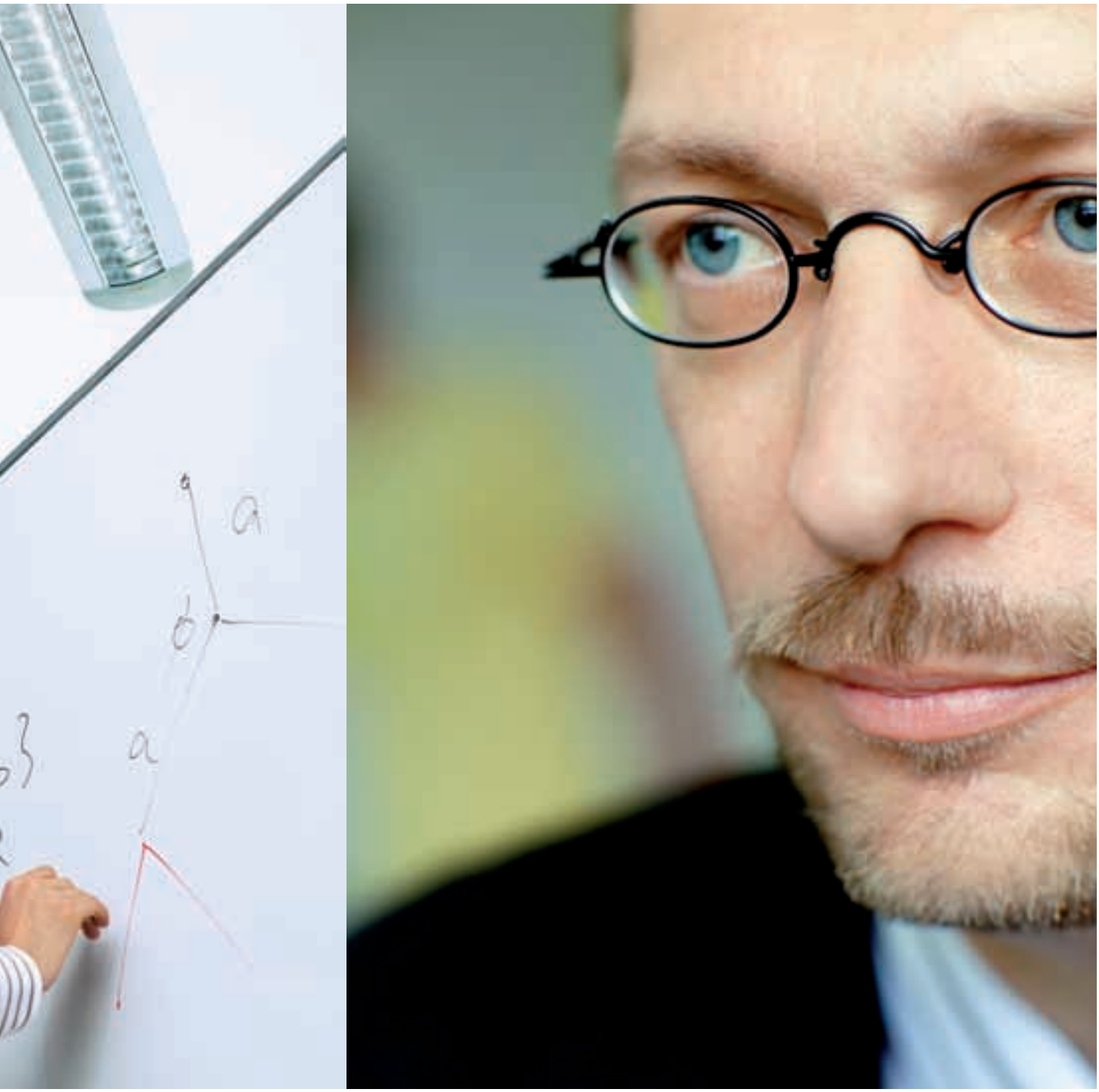
Guaranteeing the security of critical software systems is a major challenge that comprises theoretical as well as practical aspects. The chair for Modeling and Analysis of Information Systems (MAIS) develops methods and tools with solid mathematical foundations to support the precise modeling of security requirements and the reliable assurance of security guarantees.

Heiko Mantel • Die korrekte Funktionsweise von Software ist in vielen Anwendungsbereichen unabdingbar. Fehler in der Steuerung von Fahrzeugen können Leib und Leben von Passagieren bedrohen, fehlerhafte Dienste die Reputation ihrer Anbieter gefährden und Verluste von Daten enorme wirtschaftliche Schäden verursachen. Um derartige Zwischenfälle zu vermeiden, sollten kritische Aspekte von Softwaresystemen möglichst zuverlässig garantiert werden. Hierbei kann man sich der Präzision mathematisch fundierter Konzepte und Methoden bedienen.

In den letzten Jahrzehnten wurden in der Entwicklung solcher formaler Methoden in der Informatik signifikante Fortschritte erreicht. Inzwischen gibt es eine Vielzahl formaler Notationen und Kalküle, mit denen das Verhalten von Systemen elegant modelliert und kritische Systemeigenschaften präzise beschrieben und nachgewiesen werden können. Komplexe Verifikationsaufgaben lassen sich mit Hilfe von strukturierten Spezifikationen und Kompositionalitätsresultaten in weniger komplexe Probleme zerlegen, die dann mit modernen Verifikationswerkzeugen weitgehend automatisch bewiesen werden können. Durch diese Fortschritte ist sogar die Verifikation komplexer Softwaresysteme praktisch möglich geworden.

Beim Nachweis von Sicherheitsanforderungen, die die Vertraulichkeit von Informationen oder die Integrität von Daten betreffen, ergeben sich besondere Herausforderungen. Diese betreffen nicht

Heiko Mantel



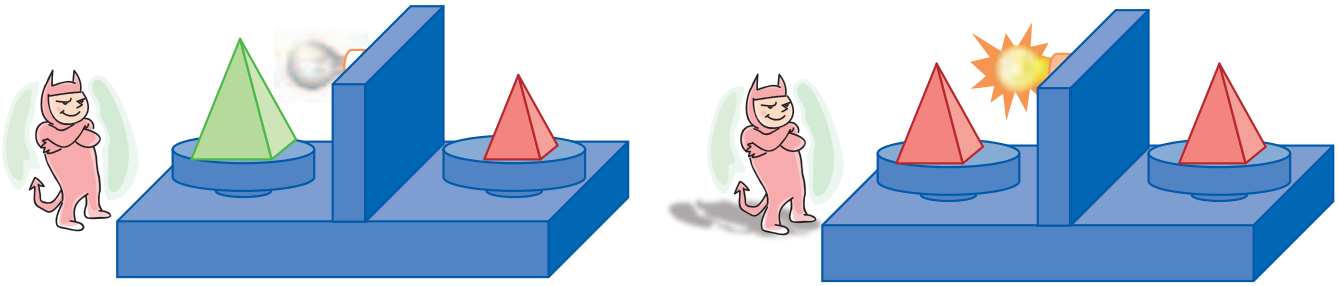


Abbildung 1
Verhinderung unerlaubter Informationsflüsse durch Noninterference

Noninterference
Bei der Geheimhaltung von Daten reicht es nicht aus, die direkte Weitergabe dieser Daten zu verhindern. Es muss auch ausgeschlossen werden, dass Angreifer mit Hilfe von erhaltenen Informationen Rückschlüsse über Geheimnisse ziehen können (vgl. Abb. 1). Die Noninterference-Eigenschaft verhindert solche unerlaubten Informationsflüsse, indem sie verlangt, dass Ausgaben (links der Trennwand) an nicht vertrauenswürdige Nutzer von Geheimnissen (rechts der Trennwand) unabhängig sein müssen.

nur den formalen Nachweis kritischer Anforderungen, sondern oft auch die vorausgehende formale Modellierung.

Betrachtet man konzeptionell einfache Sicherheitsmechanismen wie Zugangs- oder Zugriffskontrollen, so ist das Vorgehen noch mit der klassischen Verifikation der Korrektheit von Algorithmen vergleichbar. Aber schon bei der Verifikation kryptographischer Protokolle treten Besonderheiten auf. Neben den durch ein Protokoll vorgegebenen Aktionen müssen auch alle erdenklichen Aktionen potenzieller Angreifer modelliert und bei der Verifikation berücksichtigt werden. Betrachtet man noch komplexere Softwaresysteme, so kann bereits die angemessene Formulierung von Sicherheitsanforderungen zu einer nicht trivialen Herausforderung werden.

Kritische Eigenschaften sollten sowohl präzise als auch möglichst verständlich beschrieben werden, um überzeugende Sicherheitsgarantien zu erhalten. Dass die Erreichung dieser beiden Ziele nicht einfach ist, wird offensichtlich, wenn man typische Anfragen der Sicherheitsinfrastruktur betrachtet. Selbst wenn die korrekte Funktionsweise einer Zugriffskontrolle zweifelsfrei erwiesen ist, sind Aussagen wie *„Das Programm Finanzverwaltung versucht, auf das Internet zuzugreifen (Ziel-IP: 141.90.10.11).“* in ihren Folgen kaum überschaubar. Beispielsweise bleibt unklar, in welcher Weise das Internet durch das Programm verwendet werden soll, welche Informationen an andere weitergegeben werden und inwieweit erhaltene Nachrichten Veränderungen der eigenen Daten bewirken. Derartige Formulierungen von Sicherheitsaspekten bieten daher keine adäquate Basis, um Anfragen wie *„Wollen Sie diesen Zugriff erlauben?“* fundiert zu beantworten.

Ein Nutzer ist vor allem daran interessiert, was nach einer Zustimmung mit seinen Daten und Ressourcen passieren wird. Mit den Termini und der Funktionsweise technischer Sicherheitsmechanismen sollte er möglichst wenig behelligt werden. Daher wären Formulierungen, wie *„Das Programm Finanzverwaltung möchte private Daten aus dem Verzeichnis STEUERERKLAERUNG-2010 versenden. Diese Daten werden ausschließlich an die Ziel-IP: 141.90.10.11 (Finanzamt) übertragen. Weitere private Daten werden nicht übertragen. Wollen Sie diese Übertragung erlauben?“* deutlich geeigneter, um eine fundierte Entscheidung zu treffen. Weiterhin wäre es wünschenswert, wenn Antworten auf solche Detailentscheidungen in möglichst vielen Fällen aus zuvor angegebenen Sicherheitsbedürfnissen automatisch abgeleitet werden könnten. Zur Erreichung beider Ziele ist es notwendig, die Informationsflüsse und Ressourcennutzungen während einer Programmausführung präzise zu charakterisieren. Allerdings ist die formale Modellierung dieser Aspekte oft nicht einfach.

Um überzeugende Sicherheitsgarantien zu erhalten, ist es außerdem notwendig, dass diese zuverlässig nachgewiesen werden. Aktuelle Forschungsarbeiten in den Bereichen Informationsflusskontrolle und Nutzungskontrolle gehen die Problematik der adäquaten Modellierung und der Verifikation von Sicherheitsaspekten an.

Bereits Anfang der 80er Jahre wurde die Noninterference-Eigenschaft als formal definiertes Kriterium für die Sicherheit des Informationsflusses in

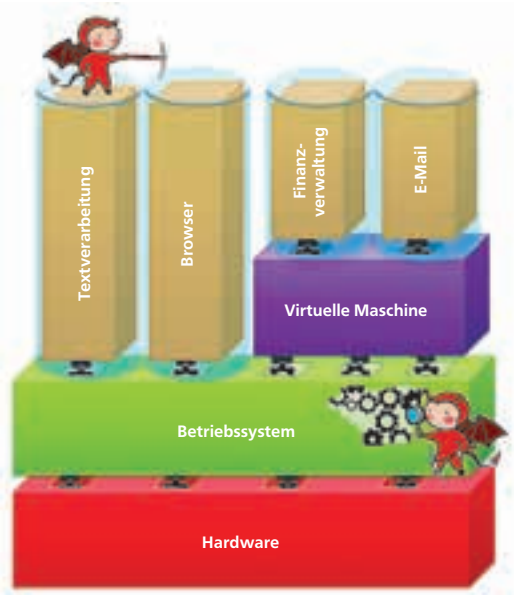


Abbildung 2
Auswirkungen von Sicherheitslücken in tieferen Systemschichten

Verdeckung von Sicherheitslücken durch Abstraktion
Selbst wenn formal nachgewiesen wurde, dass ein Anwendungsprogramm keine Schwachstellen aufweist, die ein Angreifer ausnutzen könnte, besteht die Gefahr, dass diese Anwendung über Schwachstellen in unterliegenden Systemschichten (beispielsweise im Betriebssystem) angegriffen wird. Solche Schwachstellen sind durch eine Analyse des Anwendungsprogramms alleine nicht erkennbar, da sie durch die Abstraktionen der Schichtenarchitektur verdeckt werden.Derartige Probleme sind nicht auf Schichtenarchitekturen beschränkt, sondern können generell bei der Verwendung von Abstraktionen auftreten, also beispielsweise auch in einer schrittweisen Softwareentwicklung.

einem System vorgeschlagen. Diese Eigenschaft hat sich inzwischen als Basis für präzise, formale Modellierungen von Informationsflusssicherheit etabliert.
Noninterference schließt nicht nur die unmittelbare Weitergabe vertraulicher Daten an nicht vertrauenswürdige Nutzer aus, sondern auch die Möglichkeit, dass Angreifer aus erhaltenen Daten vertrauliche Informationen errechnen können. Die zu Grunde liegende Idee ist, dass Ausgaben, die völlig unabhängig von Geheimnissen sind, einem Angreifer nicht helfen können, Informationen über diese Geheimnisse zu gewinnen. Diese Forderung führt zu sehr überzeugenden Sicherheitsgarantien, ist allerdings auch sehr restriktiv.
Manchmal ist es wünschenswert oder sogar unabdingbar, dass einige Informationen über vertrauliche Daten bekannt werden. Beispielsweise sollten Filme oder Lieder in einem Online-Shop für Nutzer vor einer Bestellung bzw. Bezahlung nicht zugänglich sein, danach aber schon.
Eine solche Deklassifikation von Informationen, die ursprünglich als vertraulich eingestuft waren, ist auch bei einer Authentifizierung unumgänglich. Aus der Reaktion auf eine Passworteingabe wird offensichtlich, ob das richtige Passwort eingegeben wurde, also ein Rückschluss auf das im System gespeicherte Passwort möglich wird. Deklassifikation muss auch zugelassen werden, wenn die Übertragung von geheimen Daten über ein offenes Netzwerk nach einer kryptographischen Verschlüsselung erlaubt werden soll.
Um die kontrollierte Preisgabe von Geheimnissen in solchen Fällen zuzulassen, muss die ursprüngliche Definition von Noninterference abgeschwächt werden. Die Entwicklung von Varianten der Noninterference-Eigenschaft, die kontrollierte Deklassifikation unterstützen, spielt für die praktische Verwendung eine zentrale Rolle, ist aber immer noch ein aktuelles Forschungsproblem. Weitere Varianten der Noninterference-Eigenschaft werden erforscht, um gängige Programmierparadigmen besser zu unterstützen. Vor allem die angemessene Behandlung nebenläufiger und verteilter Programmausführung stellt hierbei eine Herausforderung dar.
In diesen Bereichen konnte das Fachgebiet MAIS in den letzten Jahren einige vielversprechende Fortschritte erzielen. Beispielsweise wurden Varianten der Noninterference-Eigenschaft entwickelt, die eine flexible Kontrolle der Deklassifikation ermöglichen bzw. nebenläufige und verteilte Programmausführungen unterstützen. Neben der Entwicklung adäquater Modellierungen von Sicher-

Fachgebiet Modellierung und Analyse von Informationssystemen (MAIS)
Prof. Dr.-Ing. Heiko Mantel
Tel. 06151/16-6651
E-Mail: mantel@mais.informatik.tu-darmstadt.de
www.mais.informatik.tu-darmstadt.de

Abbildung 3
Durchsetzung von Sicherheitsanforderungen durch Kapselungen (links: gewöhnlicher Dienst, rechts: gekapselter Dienst)

heitsanforderungen ist auch die Entwicklung von Techniken und Werkzeugen, die die Verifikation von Informationsflusssicherheit erleichtern, ein aktueller Forschungsgegenstand. Die Erforschung von Ansätzen zur Informationsflusskontrolle wird in den nächsten Jahren durch das von der DFG geförderte Schwerpunktprogramm „Reliably Secure Software Systems“ viele neue Impulse erhalten.

Eine weitere konzeptionelle Herausforderung beim Nachweis von Sicherheitsgarantien entsteht durch die Verwendung von Abstraktionen, die für eine Verifikation komplexer Systeme unabdingbar ist (vgl. Abbildung 2). In der Informatik übliche und weit erprobte Abstraktionstechniken sind für Sicherheitsaspekte problematisch. Selbst wenn man bewiesen hat, dass alle Informationsflüsse innerhalb eines Anwendungsprogramms den Sicherheitsbedürfnissen entsprechen, können vertrauliche Daten durch Informationslecks in tiefer liegenden Systemschichten bekannt werden. Ein eng verwandtes Problem tritt bei der Verwendung von abstrakten Spezifikationen in einer schrittweisen Softwareentwicklung auf. Die Abstraktion von technischen Details ist für Spezifikationen hilfreich, kann aber sowohl Möglichkeiten zur Kommunikation zwischen Angreifern (sogenannte verdeckte Kanäle) als auch zur unbeabsichtigten Preisgabe von Geheimnissen (sogenannte Seitenkanäle) bei einer Sicherheitsanalyse verdecken. Um diese Problematiken beherrschbar zu machen, werden im Fachgebiet MAIS im Rahmen der durch die DFG geförderten Nachwuchsgruppe „Formal Methods for Security Engineering“ Techniken zur schrittweisen Entwicklung sicherheitskritischer Software und zur Analyse verdeckter Kanäle entwickelt. Techniken und Werkzeuge zur Entdeckung und Entfernung von Seitenkanälen in kryptographischen Algorithmen, die durch Unterschiede in der Laufzeit entstehen, werden im Rahmen von CASED entwickelt.

Erhält ein Softwaresystem auch während der Ausführung vertrauliche Eingaben, so ist es oft sinnvoll, die statische Sicherheitsanalyse des Programmcodes um eine dynamische Analyse zu ergänzen. Das Konzept der Sicherheitsautomaten erlaubt eine flexible Beschreibung von Komponenten zur Überwachung von Sicherheitseigenschaften zur Laufzeit eines Programms. Sicherheitsbedürfnisse werden als Richtlinien formuliert, deren Einhaltung der Sicherheitsautomat während der Programmausführung überwacht. Sollte die Verletzung einer Sicherheitsrichtlinie bevorstehen, so wird die Programmausführung gestoppt. Werden alle Sicherheitsbedürfnisse durch Sicherheitsrichtlinien erfasst, so kann auf eine statische Sicherheitsanalyse des Programms verzichtet werden.

Der Sicherheitsautomat realisiert dann eine Kapselung, die die Umgebung vor Fehlverhalten des Programms und das Programm vor Fehlverhalten nach fehlerhaften Eingaben der Umgebung bewahrt. Um die Anwendungsmöglichkeiten solcher Kapselungen zu erweitern, wurden verschiedene Varianten der Ende der 90er Jahre vorgeschlagenen Definition von Sicherheitsautomaten entwickelt. Editierautomaten erlauben zum Beispiel neben dem Anhalten des Programms im Fehlerfall auch das Unterdrücken oder Abändern einzelner Aktionen des Programms. Das im Rahmen von CASED entwickelte Konzept der Dienstautomaten zielt auf die dynamische Überwachung von Diensten in verteilten Systemen. Dezentrale Monitorkomponenten kommunizieren und kooperieren miteinander. Hierdurch wird eine flexible und zuverlässige Kontrolle der Nutzung sicherheitskritischer Daten und Ressourcen unterstützt, die der modularen und verteilten Architektur moderner dienstorientierter Systeme Rechnung trägt.

Heiko Mantel ist seit 2007 Professor an der TU Darmstadt. Er leitet das Fachgebiet Modellierung und Analyse von Informationssystemen (MAIS) im Fachbereich Informatik. Er koordiniert das DFG-Schwerpunktprogramm „Reliably Secure Software Systems“ und ist zudem Principal Investigator des LOEWE-Zentrums CASED.