Modellbasierte Softwaretechniken für sichere Systeme

Prof. Dr. Jan Jürjens

http://jurjens.de/jan/teaching/ws09/mbse

TU Dortmund, Fak. Informatik, LS 14

technische universität dortmund

Fraunhofer ISST

http://jurjens.de/jan

Diese Vorlesung

Einführung in Modell-basiertes Sicherheits-Engineering

- Überblick IT-Security: Was ist zu schützen ? Was sind die Angriffe ?
- Sicherheitsrelevante Workflows / Geschäftsprozesse, Bedrohungsanalysen
- Überblick IT-Sicherheitsmaßnahmen, Daten- und Dokumentensicherheit (z.B. Digitale Signatur)
- IT-Zugangsschutz, Netzwerksicherheit
- Modell-basierte Entwicklung mit UML
- Werkzeuge
- Industrielle Anwendungen (Biometrie, Smart-cards, …)

Organisatorisches

Diplom: Schwerpunkte 1 (Software-Konstruktion) und 5 (Sicherheit und Verifikation).

Montags 14.00-16.00, GB IV Raum 322

Übungen: Jan Jurjens, Martin Ochoa.

- Montags 12.00-14.00, GB IV Raum 322
- Abgabe in der Uebung oder per e-mail an: martin.ochoa@cs.tu-dortmund.de jeweils bis folgenden Montag. Aufgabenstellungen auf Vorlesungs-Folien.

Scheinkriterium: 50% Aufgabenpunktzahl, aktive Teilnahme an Übungen, mdl. Test

http://jurjens.de/jan/teaching/ws09/mbse
(user mbse09 , password wjdkhora)

Mailing list.

Modul-Beschreibung (Master), Ausschnitt

Modulkatalog Master-Studiengänge Informatik
Modul INF-MSc-420: Modellbasierte Softwaretechniken für sichere Systeme
MA-Studiengänge: Informatik, Angewandte Informatik
Turnus: Jedes WS. Dauer: 1 Sem.. Studienabs.: 2.-3. Sem., Credits: 6, Aufwand: 180 h (60/120)
Modulstruktur:

Nr.	Element / Lehrveranstaltung	Typ	Credits	SWS
1	Modellbasierte Softwaretechniken für sichere Systeme	V	3	2
1	Übungen zu Modellbasierte Softwaretechniken für sichere Systeme	Ü	3	2

Lehrveranstaltungssprache: Deutsch und/oder Englisch
Prüfungen

- Modulprüfung: Klausur oder mündliche Prüfung
- Studienleistungen: 50% der Aufgabenpunktzahl, aktive Teilnahme an Uebungen (beides Voraussetzung fuer die Zulassung zur Pruefung), Pruefung.
- Prüfungsformen und -leistungen: Modulprüfung

7 Teilnahmevoraussetzungen

- Basismodul aus dem Forschungsbereich A (Software, Sicherheit und Verifikation)

Modultyp und Verwendbarkeit des Moduls

- Vertiefungsmodul in den Masterstudiengängen Informatik und Angewandte Informatik
- Forschungsbereich: Software, Sicherheit und Verifikation

Werbe-Block :-)

Ebenfalls relevant und interessant:

- Blockseminar aufbauend auf diese Vorlesung: Vorbesprechung am 9. Nov. (hier in dieser Vorlesung), Blockseminar am 8./9. Maerz: http://jurjens.de/jan/teaching/ws09/mbse-sem
- Seminar “Architektur- und Geschäftsprozess-Modellierung”: Montags im Anschluss an diese Vorlesung, Raum 318 (Themenvergabe 26.10., Start 9.11.): http://jurjens.de/jan/teaching/ws09/agm-sem
- Blockseminar “E-Commerce Systeme: Architekturen und Anwendungen”: Vorbesprechung am Mo 2.11., 13.00-14.00 Uhr (Raum 322), Block-seminar am 1./2. Maerz: http://jurjens.de/jan/teaching/ws09/ecom-sem
- Betreuung von Bachelor- / Master- / Diplomarbeiten z.B. zu Themen dieser Vorlesung bzw. der Seminare, auch in Zusammenarbeit mit Fraunhofer ISST: http://jurjens.de/jan/jobs/hiwis.html
- SHK-Stellen zu vergeben: http://jurjens.de/jan/jobs/hiwis.html

Literatur

Allgemeiner Hintergrund:

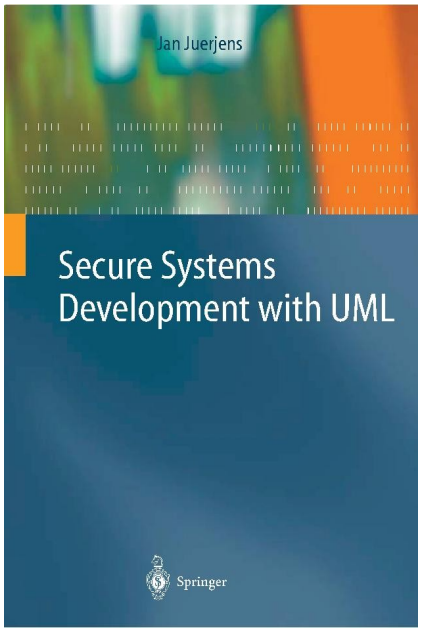
Gasser: Building a Secure Computer System (1988) [UB: -]

Anderson: Security Engineering (2001) [UB-Inf: 3378/Ande, 3384/Ande]

Diese Vorlesung basiert auf:

Jan Jürjens, Secure Systems Development with UML, Springer-Verlag 2005, cf http://umlsec.de

TU-Bibliothek: Signaturen L Sr 531 bis L Sr 531+4



1 Einführung

IT-Sicherheit

Wirtschaft, Unternehmen und Gesellschaft hängen zunehmend ab von Computernetzwerken für Kommunikation, Finanzen, Energieversorgung, Transport...

Angriffe können großen finanziellen Schaden verursachen.

Vernetzte Systeme können anonym und aus der Entfernung angegriffen werden.

Computersysteme müssen also sicher sein.

Problem: Sicherheit

Sicherheit (Security): Schutz von Daten oder Systemen gegen mutwillige Angriffe. Inhärent schwierig (zielorientierter Angreifer). Beispiel (1997):

NSA hacker team bricht in U.S. Department of Defense Computer und U.S. Strom-versorgungssystem ein. Demonstriert Strom- und Notrufausfälle in Washington, D.C..

Beispiele für Sicherheitsvorfälle

- Einbruch in die Website SalesGate.com, Diebstahl von 3.000 Kundendateien (z.B. Kreditkartennummern). Z.T. im Internet veröffentlicht.
- Unkontrollierte Weiterleitung persönlicher Informationen aus Hersteller-Sites (z.B. Finanzrechenprogramme auf Intuit) zu Anzeigen-Sites (wie DoubleClick) ohne Wissen der Benutzer und oder von Intuit.
- Februar 2000: massive Denial-of-Service Angriffe.

[Schneier: Secrets & Lies]

Softwareschwächen (9.11.-1.12.04)

- Microsoft schließt das IFrame-Loch (1.12.2004, ju). Mit einem überraschenden Update beseitigt Microsoft das IFrame-Problem des Internet Explorer 6.0.
- Buffer Overflow in Suns Ping-Befehl (1.12.2004, dab). Sun weist in einem Advisory auf einen Buffer Overflow im Ping-Befehl hin, mit dem angemeldete Nutzer unter Umständen ihre Zugriffsrechte erhöhen können.
- Cross-Site-Scripting-Schwachstelle in Linux-Firewall IPCop (1.12.2004, dab). Durch eine Cross-Site-Scripting-Schwachstelle in IPCop kann ein Angreifer das Authentifizierungscookie des Administrators stehlen und sich damit später ohne Kenntnis des Passwortes an der Firewall anmelden.
- Server des CCC gehackt (29.11.2004, pab). Spanische Hacker sind in die Server des Chaos Computer Clubs eingedrungen und haben unter anderem die Registrierungsdaten vom CCC-Camp 2003 veröffentlicht.
- Windows-Namensdienst verwundbar (29.11.2004, ju). Im Windows-Namensdienst WINS gibt es einem Advisory von Nicolas Waisman zufolge eine Schwachstelle, über die ein Angreifer beliebigen Code einschleusen und ausführen kann.
- SQL-Injection-Lücke in PHPNews beseitigt (25.11.2004, dab). In Version 1.2.4 der Board-Software PHPNews wurde eine SQL-Injection-Schwachstelle beseitigt.
- Lücke in Suns Java Plug-ins gewährt Zugriff auf das System (23.11.2004, dab). Durch einen Fehler in Suns Java Plug-ins für Browser können Angreifer mit präparierten Java Applets aus der Sandbox ausbrechen und die Kontrolle über den Rechner erlangen. Betroffen sind alle Browser, die Suns Plug-in einsetzen.
- Vergiftete Websites [Update] (22.11.2004, ju). Langsam lichtet sich der Nebel um die IFrame-Attacken vom Wochenende. Falk eSolutions leitete offenbar Zugriffe auf seine Ad-Server auf einen kompromittierten Server um, der Trojaner auf den Systemen der Anwender installierte.
- Zone Labs beseitigt DoS-Schwäche in Firewall-Produkten (19.11.2004, dab). Der Hersteller Zone Labs weist auf seinen Seiten auf eine Schwachstelle in seinen Firewall-Produkten hin, durch die das System zum Stillstand kommen kann.
- Samba-Entwickler schließen kritische Lücke -- ohne darauf hinzuweisen [Update] (15.11.2004, dab). Weil die Entwickler von der Ausnutzbarkeit eines Fehlers nicht überzeugt waren, beseitigten sie die Lücke, ohne darauf in einem Advisory hinzuweisen. Ein Sicherheitsexperte will aber dafür einen Exploit entwickelt haben, der Code auf dem Server ausführt.
- Angeblich zehn Sicherheitslücken in Service Pack 2 für Windows XP (12.11.2004, dab). Der Hersteller Finjan hat nach eigenen Angaben zehn gravierende Sicherheitslücken in Windows XP Service Pack 2 festgestellt. Damit sollen Hacker relativ einfach in Netzwerke eindringen und die Kontrolle über Clients gewinnen können.
- DHCP-Pakete blockieren Netzwerkschnittstellen auf Cisco-Routern (11.11.2004, dab). Cisco hat eine Schwachstelle in Geräten mit IOS-Version 12.2s gemeldet. Fehlerhafte DHCP-Pakete können die Eingangsqueue einer Netzwerkschnittstelle verstopfen, sodass der Router keine an ihn direkt gerichteten Pakete mehr annimmt.
- Update behebt Schwachstelle in Microsoft ISA und Proxy Server (9.11.2004, dab). Ein Angreifer kann einen Fehler im DNS-Cache des ISA und Proxy Server ausnutzen, um Anwender auf falsche Web-Seiten umzuleiten.
- Suns Messaging Server gewährt unautorisierten Zugriff auf Webmail-Konten (9.11.2004, dab). Die Webmail-Funktion in Suns iPlanet Messaging Server und Sun ONE Messaging Server gewährt unter besonderen Umständen Angreifern Zugriff auf Mail-Konten.
- Fehler in Ruby CGI-Modul bringt System zum Stillstand (9.11.2004, dab). Im CGI-Modul cgi.rb der objekt-orientierten Skriptsprache Ruby wurde eine Schwachstelle entdeckt, mit der Angreifer über das Netzwerk das komplette System zum Stillstand bringen können.
- Denial-of-Service-Schwachstelle in Samba-Server (9.11.2004, dab). Dateinamen mit zu vielen Wildcard-Zeichen erhöhen die Prozessorlast so stark, dass der Server nicht mehr antwortet.

[Heise security, 1.12.2004]

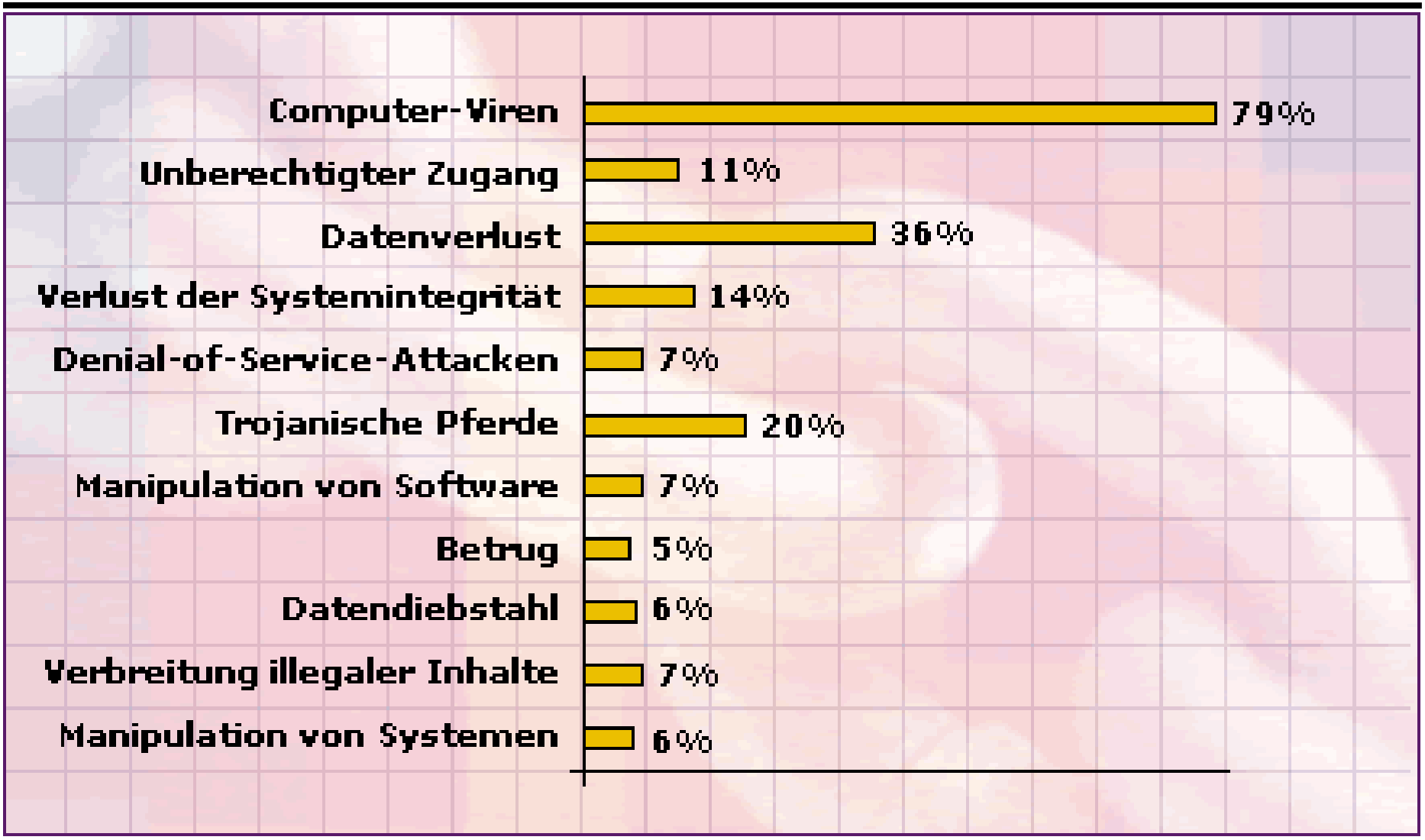
Gehackte Web-Seiten, 6.12.04 bis Mittag

Todays reported and verified attacks: 1204 of which 352 are single IP and 852 mass defacements

Time	Attacker				Domain	OS	View	
•14:58	PHTeam				....com/cgi-bin/index.cgi	FreeBSD	view | mirror	
•14:56	hackbsd crew	H	M		automondial.ro	Linux	view | mirror	
•14:53	BI0S	H			hosting2b.com	Linux	view | mirror	
•14:53	BI0S	H	M		statebase.com	Linux	view | mirror	
•14:52	Next Time	H			hightechtoys.it	Linux	view | mirror	
•14:47	Antrax	H			healthlawtoday.com	Linux	view | mirror	
•14:46	Antrax	H			mosessinger.com	Linux	view | mirror	
•14:45	DeF4x0rz Group				miawolf.net/guestbook	Linux	view | mirror	
•14:39	DeF4x0rz Group				...flats.com.br/guestbook	Linux	view | mirror	
•14:37	BI0S	H	M		psynix.com	Linux	view | mirror	
•14:37	Q8Crackers	H			groovetx.com	Linux	view | mirror	
•14:36	BI0S	H			tamingfire.com	Linux	view | mirror	
•14:35	BI0S	H			carpet24.com	Linux	view | mirror	
•14:35	NaOnaK	H			forums.deeko.com	Linux	view | mirror	
•13:51	Logicb0x		M		pcxp.piusx.net/index.htm	Win 2000	view | mirror	
•13:51	Logicb0x				...s.wnyric.org/index.htm	Win 2000	view | mirror	
•13:51	Logicb0x				...rn.k12.or.us/index.htm	Win 2000	view | mirror	
•13:50	KERANGKA|LANGIT			M	...udi.gov.cn/igenus/temp	NetBSDOpenBSD	view | mirror	
•13:50	Logicb0x				pcxp.usd262.net/index.htm	Win 2000	view | mirror	
•13:49	Next Time		M		...o.ch.it/media/next.htm	Linux	view | mirror	
•13:49	Logicb0x				pcxp.usd437.net/index.htm	Win 2000	view | mirror	
•13:48	DeF4x0rz Group			R	bodamagica.com/visitas	Linux	view | mirror	
•13:48	DeF4x0rz Group				emcorner.it/book	Linux	view | mirror	
•13:47	Logicb0x				...aschools.org/index.htm	Win 2000	view | mirror	
•13:46	SyRiaN_HacKerZ	H			syria4you.com	Linux	view | mirror	
•13:41	NaOnaK	H			phantom-legion.net	Linux	view | mirror	
•13:04	Simiens	H			nexusthegame.com	FreeBSD	view | mirror	
•13:03	Logicb0x				...ll.k12.pa.us/index.htm	Win 2000	view | mirror	
•13:02	BI0S	H			uralmebel.ru	FreeBSD	view | mirror	
•12:59	batistuta		M		moe.go.th/moego	Linux	view | mirror	
•12:59	Logicb0x				pcxp.lex2.org/index.htm	Win 2000	view | mirror	
•12:57	BI0S	H	M		bayareamarine.com	Linux	view | mirror	
•12:55	BI0S	H	M		erwinsautosales.com	Linux	view | mirror	
•12:55	BI0S	H	M		boxlaser.com	Linux	view | mirror	
•12:55	BI0S	H	M		localaw1.com	Linux	view | mirror	

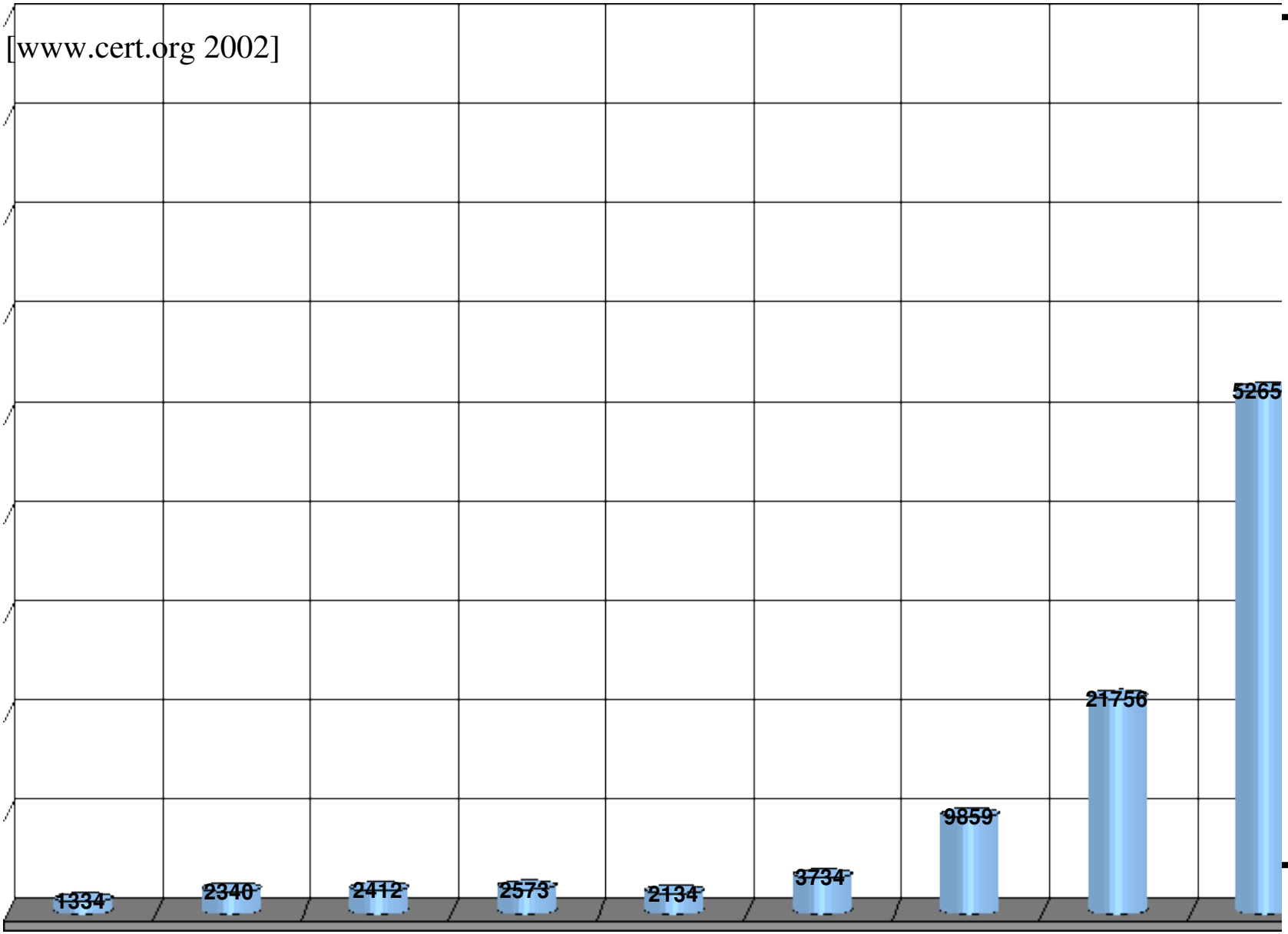
[www.zone-h.org]

Arten von Sicherheitsproblemen



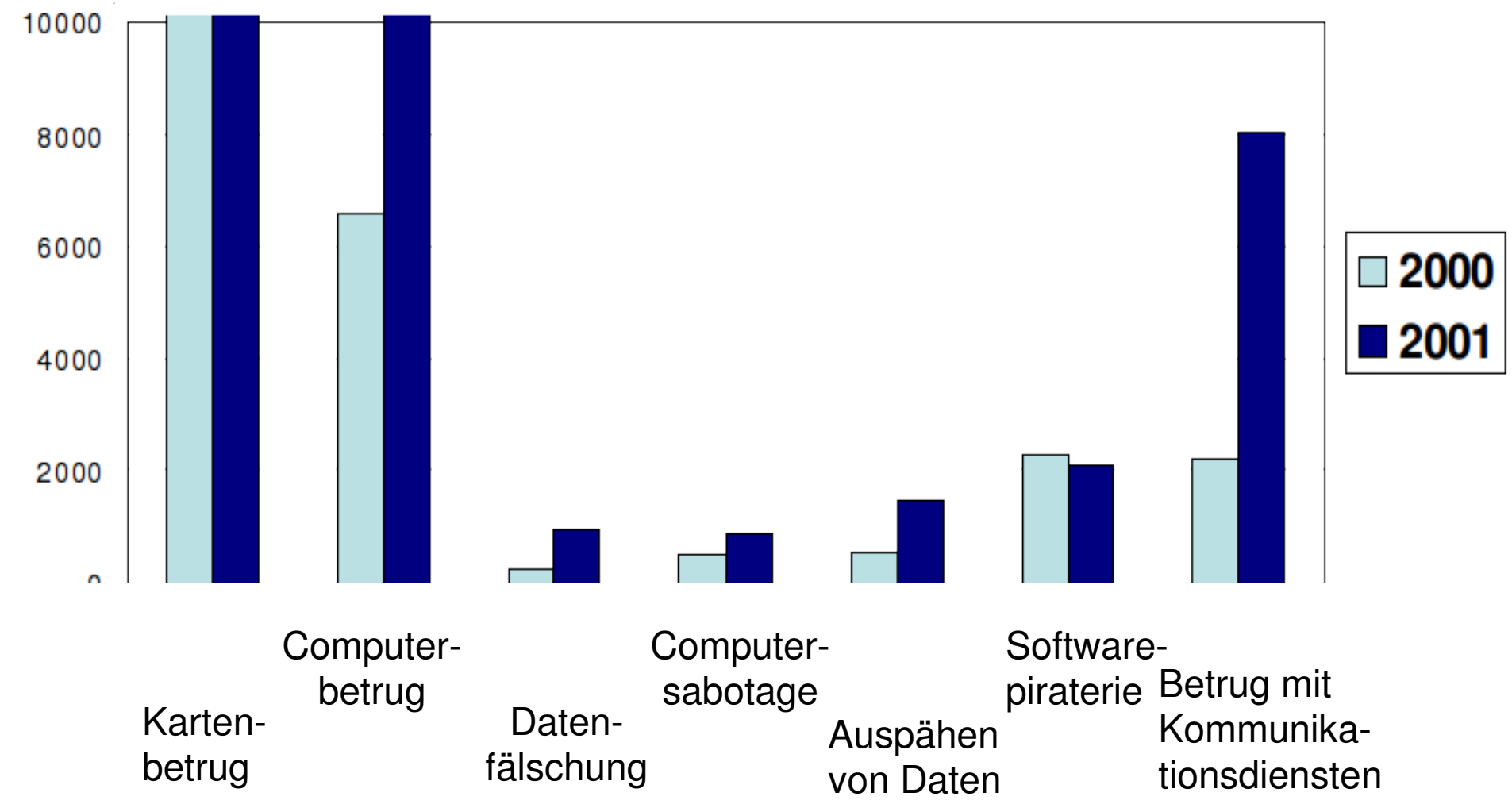
[silicon.de, 2001]

Gemeldete IT-Sicherheitsvorfälle



Kriminalstatistik

[Bundesinnenmin.]



IT-Risiken vs. KontraG/Basel II

Basel II (bis 2006): risikogerechtere Regelung der Eigenkapitalanforderungen

Genauere Analysemethoden (Kreditrisiko, operationelles Risiko, *internal-ratings-based).* Offenzulegen.

Insbesondere IT Risiken (*unexpected loss,* z.B. Virenbefall, Hackerangriff, …)

➔ modellbasierte IT-Risiko-Bewertung

Causes I

- Designing secure systems correctly is difficult. Even experts may fail:
  - Needham-Schroeder protocol (1978)
  - attacks found 1981 (Denning, Sacco), 1995 (Lowe)
- Designers often lack background in security.
- Security as an afterthought.
- Exploit information spreads quickly.
- No feedback on delivered security from customers.

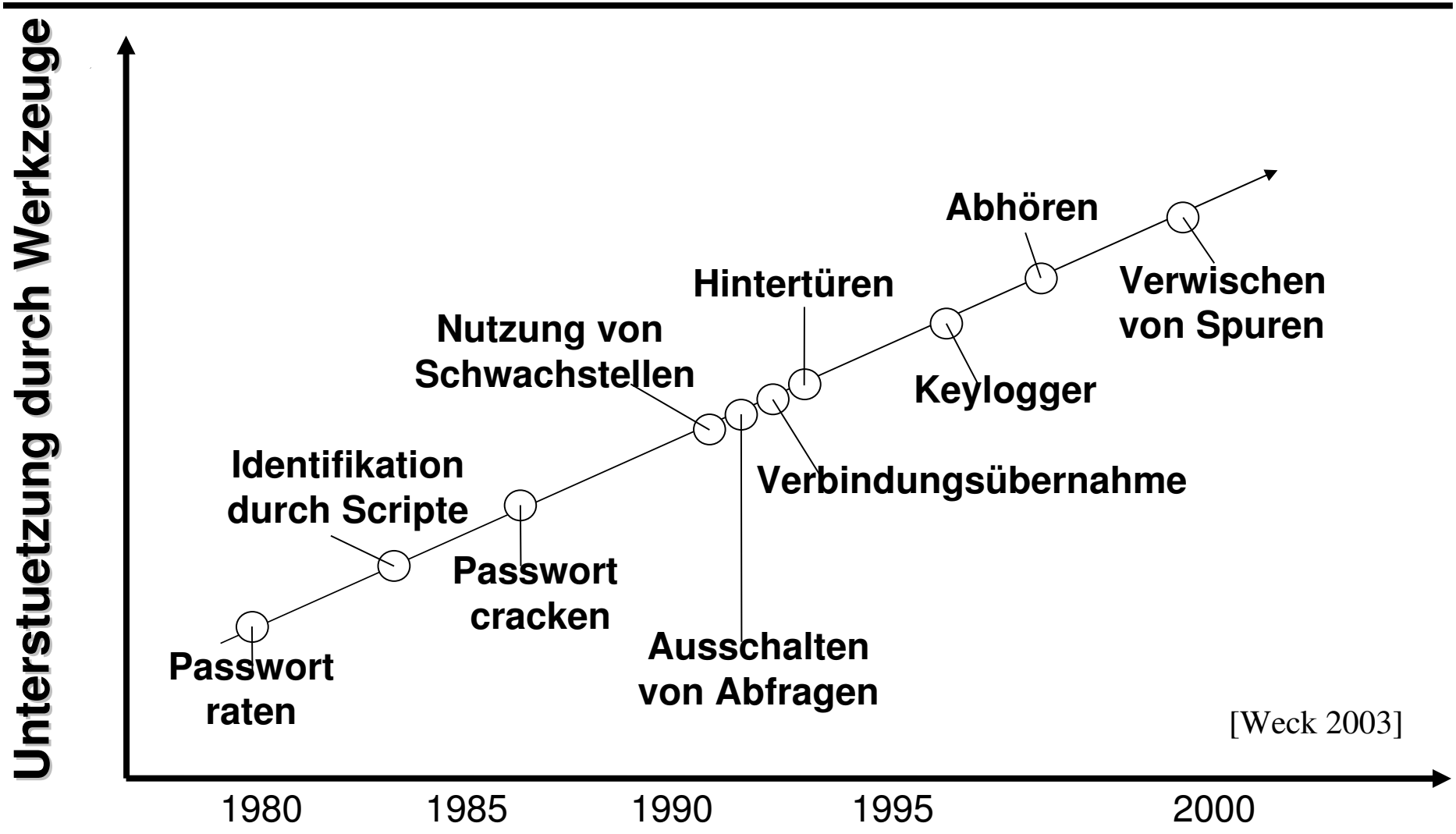
Causes II

„Blind“ use of mechanisms:

- Security often compromised by circumventing (rather than breaking) them.
- Assumptions on system context, physical environment.

„Those who think that their problem can be solved by simply applying cryptography don`t understand cryptography and don`t understand their problem“ (R. Needham).

Immer bessere Hackerwerkzeuge



[Weck 2003]

Sichere Systeme

Analyse von sicherheitskritischen Systemen schwierig (motivierter Angreifer).

Viele entwickelte und eingesetzte Systeme genügen nicht Sicherheitsanforderungen.

Sichere Produkte oft auf unsichere Weise eingesetzt.

Viele z.T. spektakuläre Angriffe.

Problem: Qualität vs. Kosten.

Holistic view on Security

„An expansive view of the problem is most appropriate to help ensure that no gaps appear in the strategy“ (Saltzer, Schroeder 1975).

But „no complete method applicable to the construction of large general-purpose systems exists yet“ - since 1975.

Software Engineering & Security

„Penetrate-and-patch“
(aka „banana strategy):

- insecure
- disruptive

Traditional formal methods: expensive.

- training people
- constructing formal specifications.

Problem: Security is Elusive

- Classical weakness in old Unix systems: “wrong password” message at first wrong letter in password. Using timing attack, reduce password space from 26^n to 26*n (n = password length)
- More recent weakness on smart-card: reconstruct secret key by timed measurement of power consumption during crypto operations

➔ How do you find these weaknesses using classical testing ?

Problem: Untrustworthy Programmer

- For security assurance, may not even trust the programmer of the code.
- May have intentionally built in back-door into code.
- May be impossible to find by random or black-box testing (e.g. hard-coded special password).
- Even worse when elusive weaknesses are used (previous slide).

➔ What is the precaution in practice?

(Usually none.)

Special Problem: Crypto

- Cryptography plays important role in many security-critical applications
- By definition, needs to be secure against brute-force attacks

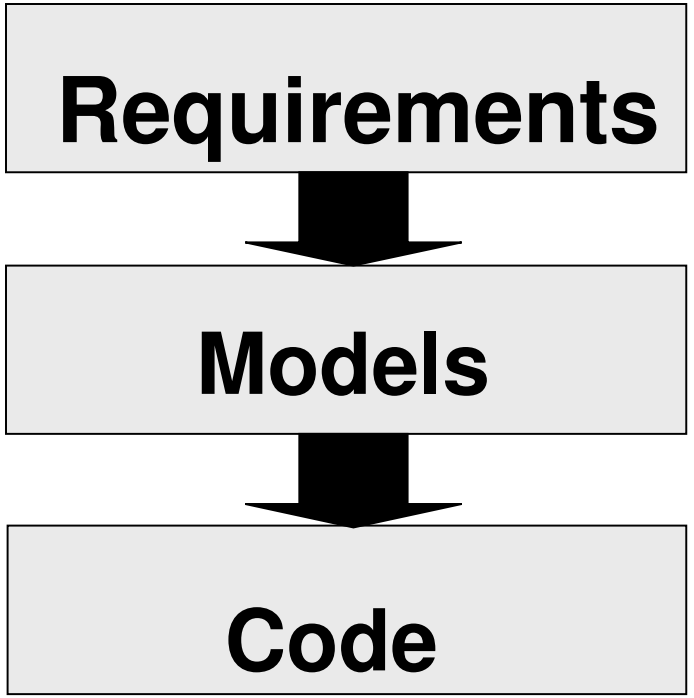
➔ Paradox: How do you get sufficient test coverage (for inputs accessible to a given attacker) of a system that needs to be secure against brute-force attacks on that input ?

(Not using classical testing.)

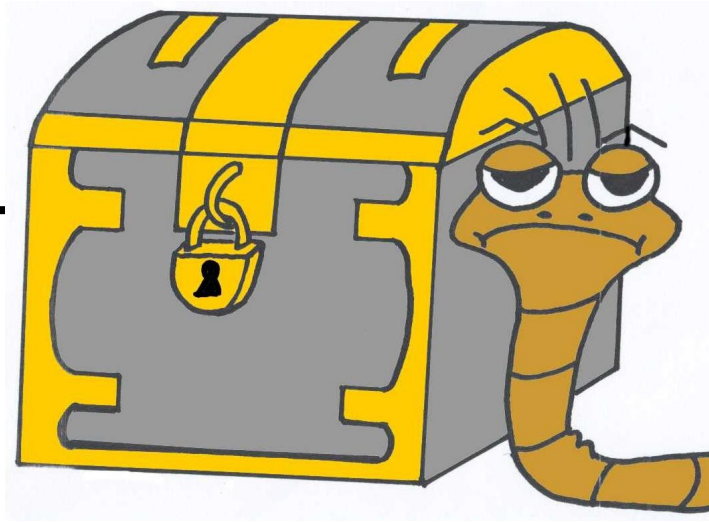
Model-based Development

Goal: easen transition from human ideas to executed systems.

Increase quality with bounded time-to-market and cost.



Security Engineering



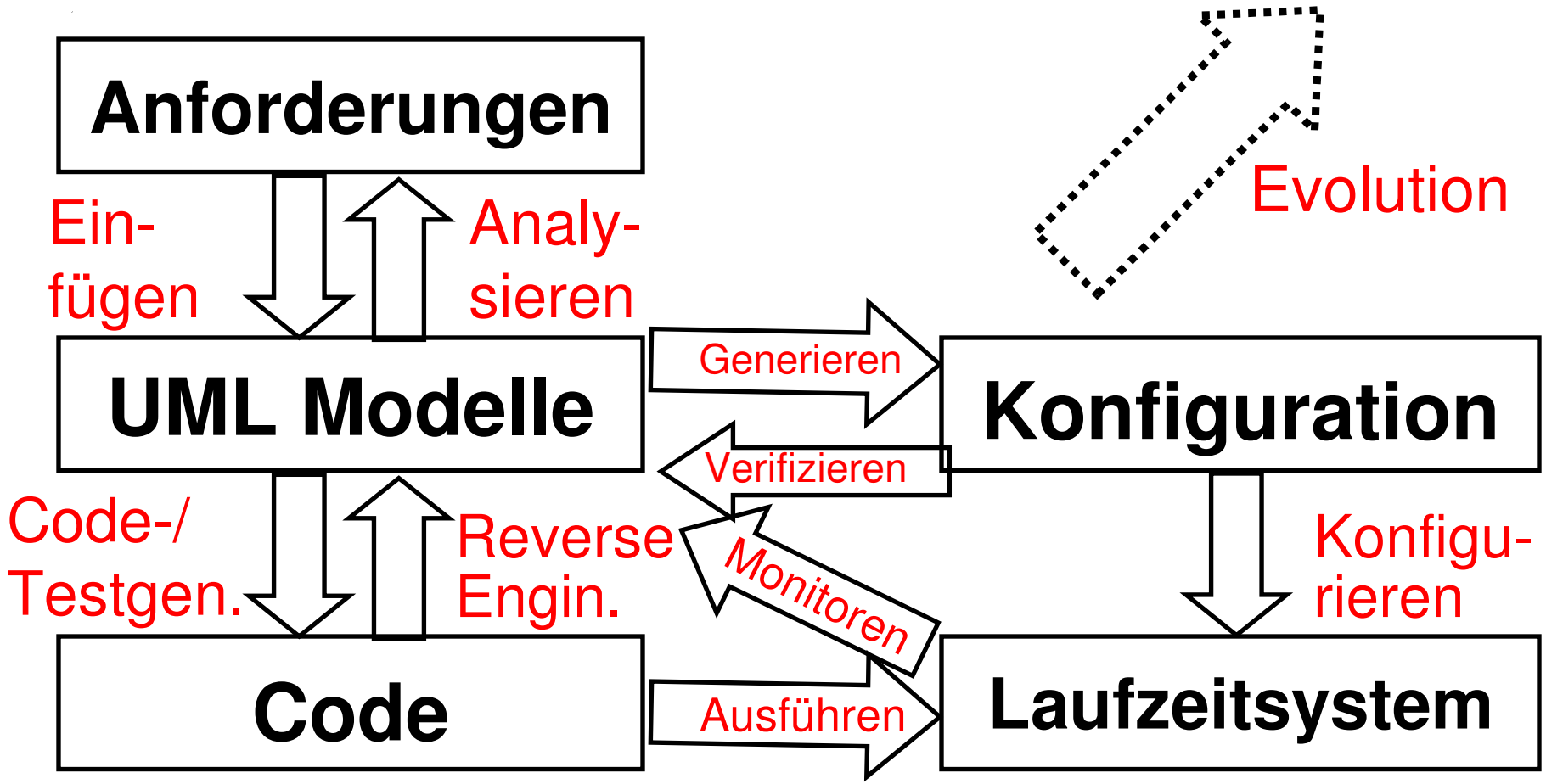
Sicherheit erhöhen bei begrenzter Zeit, Kosten.

Lösungsansatz:

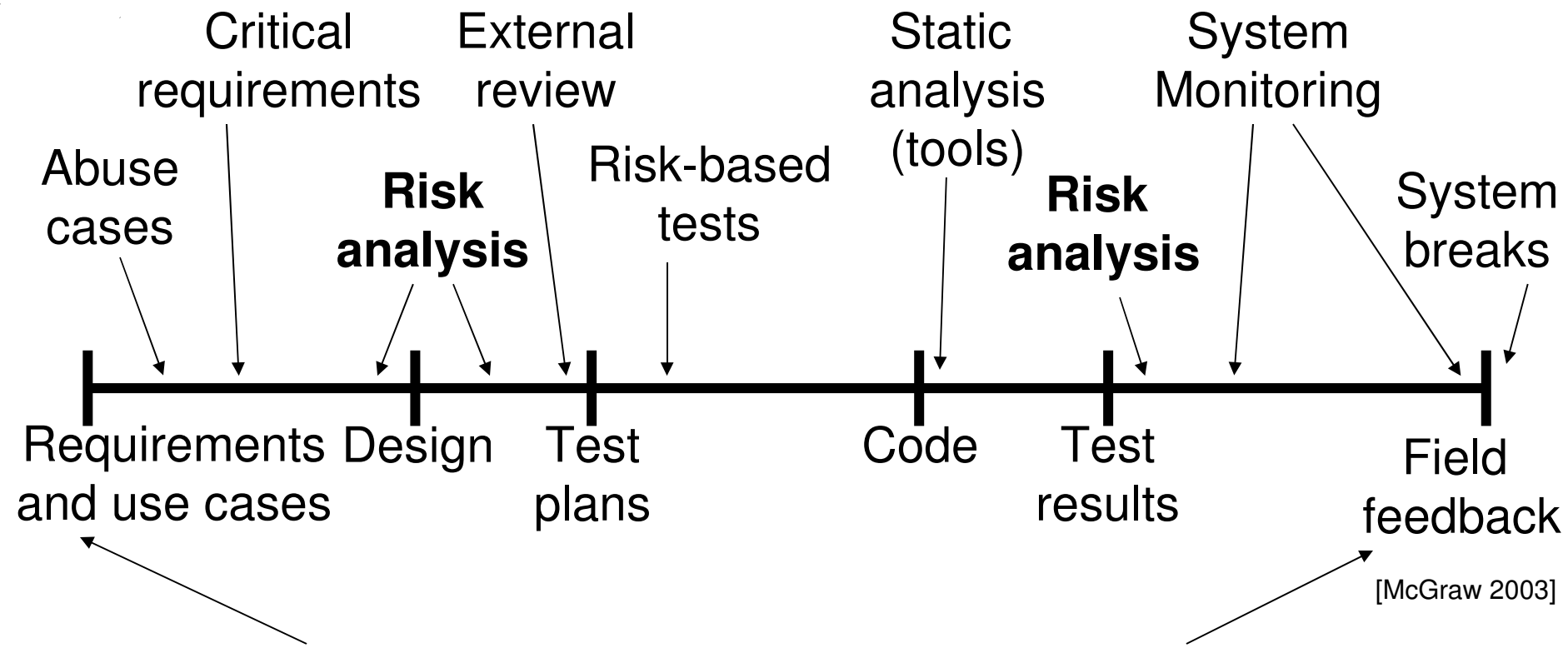
- aus Artefakten in industrieller Entwicklung und Betrieb sicherheitskritischer Software: Modelle extrahieren (UML, Quellcode, Konfigurationen)
- Werkzeugunterstützung für theoretisch fundierte effiziente (automatische) Sicherheitsanalyse

➔*Modell-basiertes Security Engineering*

Modellbasiertes Software Engineering



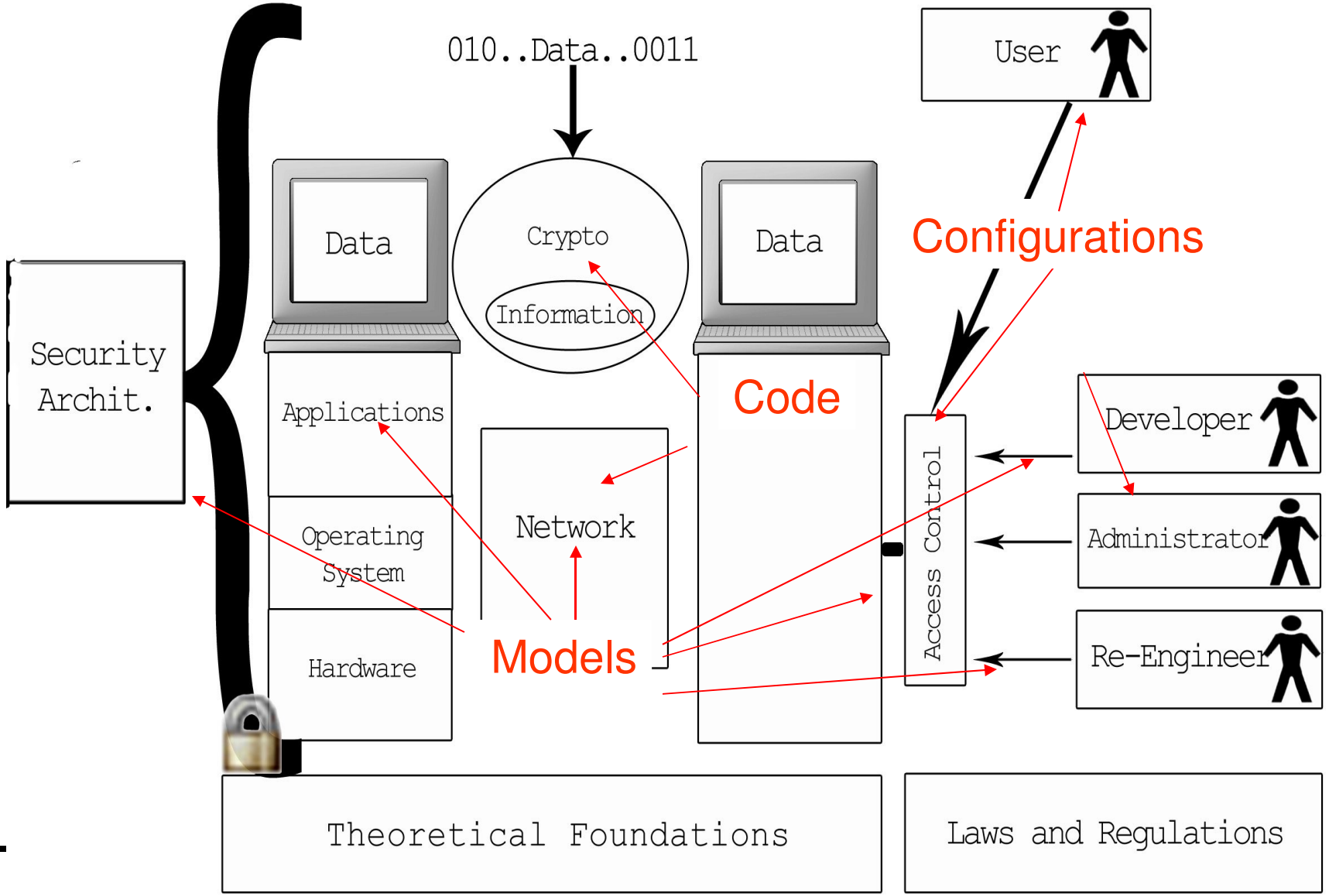
Critical System Lifecycle



[McGraw 2003]

Model-based Security Engineering

Architectural Layers



Goal: Security by design

Consider security

- from early on
- within development context
- taking an expansive view
- in a seamless way.

Secure design by model analysis.

Secure implementation by test generation.

Vorteile

- Verwendung bewährter Regeln für sichere Systeme.
- Verwendbar ohne spezielle Ausbildung.
- Berücksichtigung von Sicherheit ab Geschäftsprozessentwurf.
- Erhöht Vertrauen in Korrektheit und Vollständigkeit von Audits.
- Unterstützt Zertifizierungen.

Modellbasierte Sicherheitsanalyse

Modellbasierte Sicherheitsanalyse von Geschäftsprozessen mit der Unified Modeling Language (UML):

- Einfache, intuitive Notation
- Komfortable Werkzeugunterstützung
- Automatische Sicherheits- und Risikoanalyse der modellierten Geschäftsprozesse unter Einbezugnahme des Unternehmensumfeldes
- Automatische Checks von System-konfigurationen (z.B. SAP-Berechtigungen, …)

Einsatzfelder

- Systementwurf
  - z.B. Architekturbewertung (Beispiel: Teleworking), Plattformenwahl, Altsystemeinbindung.
- Implementierung
  - Quellcodeanalyse, Testfolgengenerierung.
- Laufender Betrieb
  - Konfigurationsmanagement, Überprüfung von Berechtigungen, Einrichtungen von Firewalls…
- Sichere Geschäftsprozesse / Behördenvorgänge
- Einsatz in Sicherheitsaudits

Why UML ?

Seemingly de-facto standard in industrial modeling.
Large number of developers trained in UML.
Relatively precisely defined (given the user community).
Many tools in development (also for code-generation, testing, reverse engineering, simulation, transformation).

Using UML

Goal: transport results from formal methods to security practice

Enable developers (not trained in formal methods) to

- check correctness of hand-made security protocols
- deploy protocols correctly in system context
- allow to analyze larger system parts beyond protocols

UMLsec: Goals

Extension for secure systems development.

- evaluate UML specifications for weaknesses in design
- encapsulate established rules of prudent secure engineering as checklist
- make available to developers not specialized in secure systems
- consider security requirements from early design phases, in system context
- make certification cost-effective

UMLsec: How

Recurring security requirements, adversary scenarios, concepts offered as stereotypes with tags on component-level.

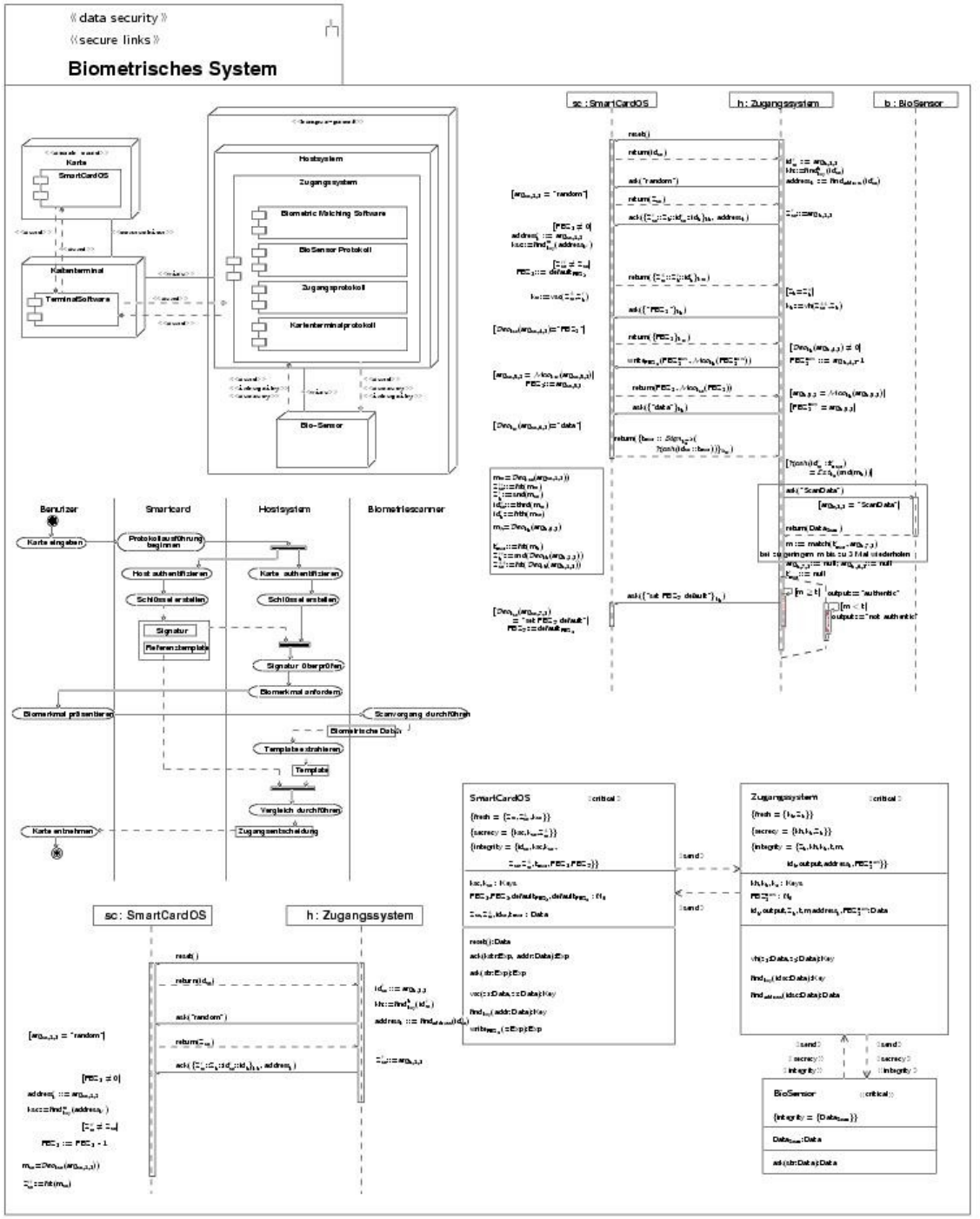
Use associated constraints to verify specifications using automated theorem provers and indicate possible weaknesses.

Ensures that UML specification provides desired level of security requirements.

Link to code via round-trip engineering etc.

Example: Biometric authentication system in industrial development.

Secure ?



Tasks

- Adapt UML to critical system application domains.
- Correct use of UML in the application domains.
- Conflict between flexibility and unambiguity in the meaning of a notation.
- Improving tool-support for critical systems development with UML.

Some Open Problems

Secure systems out of (in)secure mechanisms.

Security as pervasive property: vs. dependability, program analysis, formal methods, software engineering, programming languages, compilers, computer architectures, operating systems, reactive systems, …, …

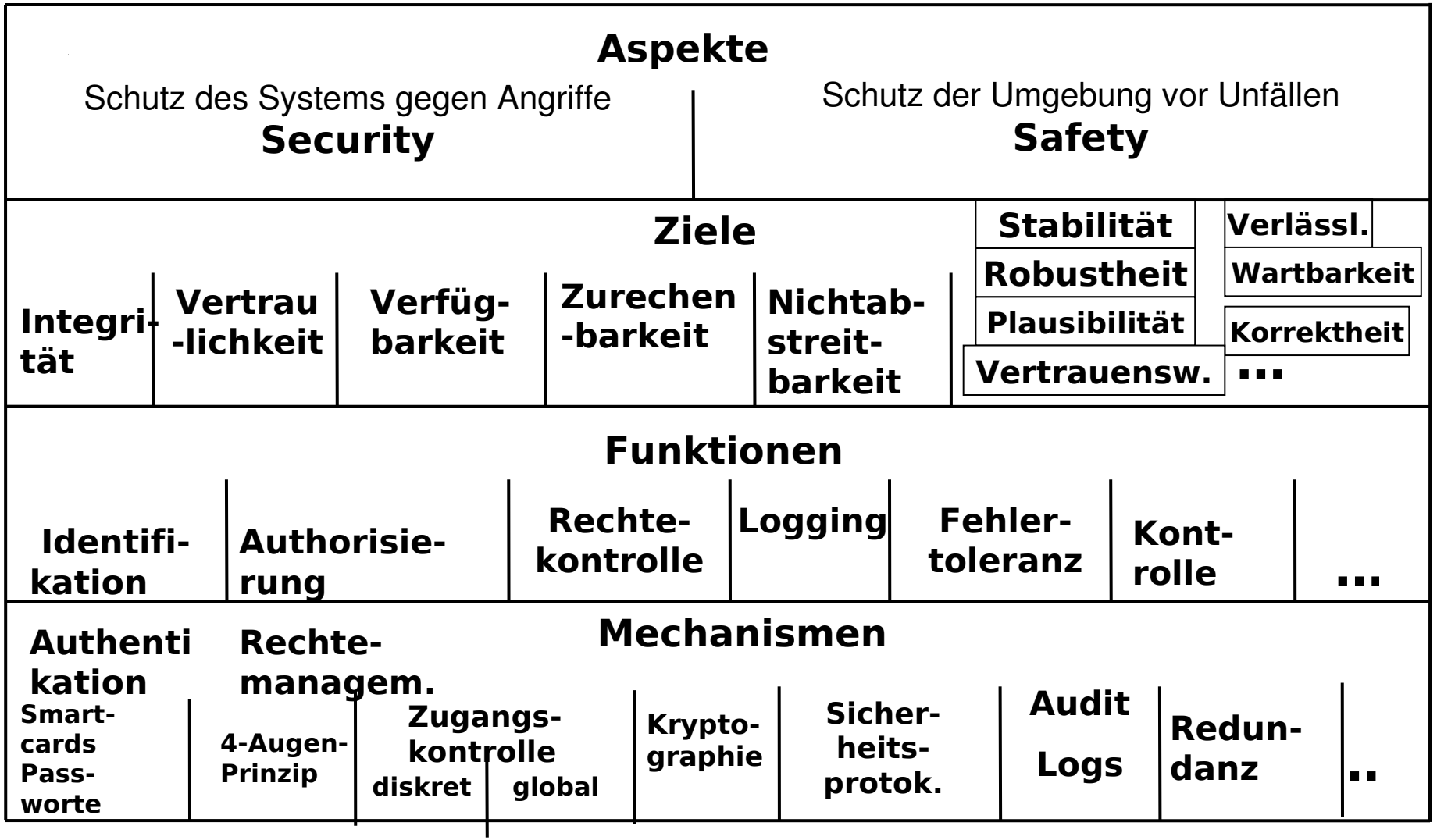
Problem: no integration / coherence.

How to put all this stuff together in a water-tight way within security engineering approach ?

Necessary for security (attacks on boundaries between views / aspects / levels …).

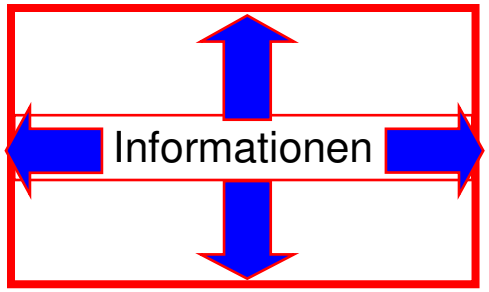
Sicherheitsanforderungen

[Weck 2003]

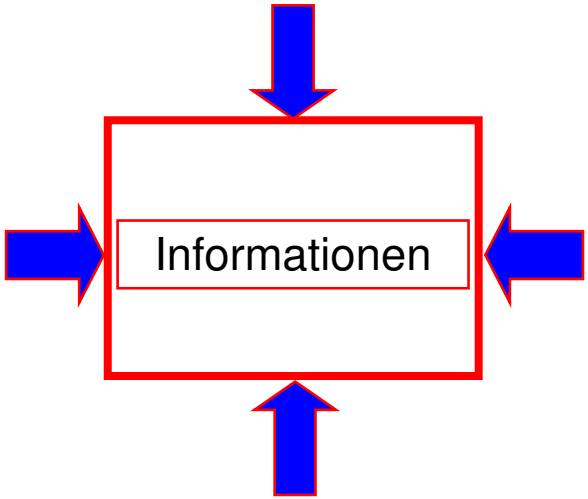


Sicherheitsanforderungen I

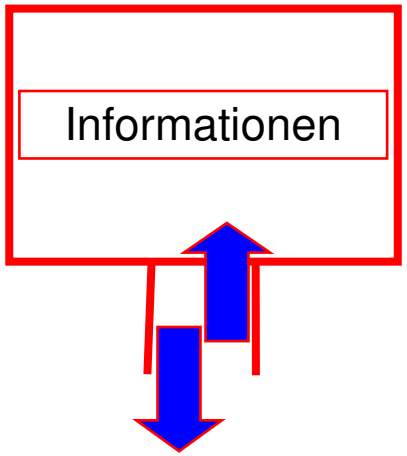
Vertraulichkeit



Integrität

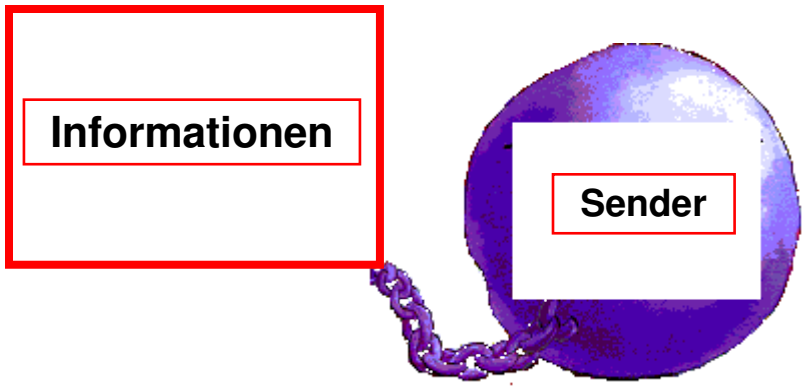


Verfügbarkeit

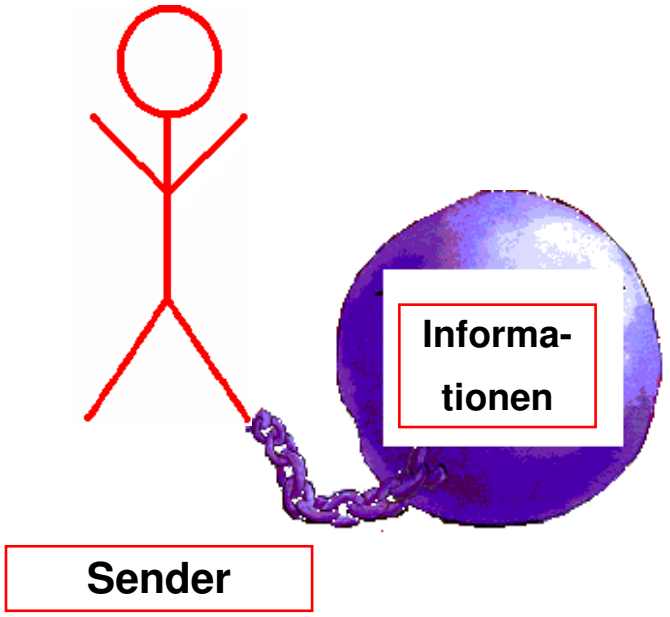


Sicherheitsanforderungen II

Authentizität



Nichtabstreitbarkeit



Gibt noch weitere: Anonymität von Benutzern, Nicht-Duplizierbarkeit von elektronischem Geld, …

Aufgabe 1

a) Die obengenannten Sicherheitsanforderungen sind im Allgemeinen unabhängig voneinander. Finde je 3 verschiedene Beispiele in der physikalischen oder digitalen Welt, sodass in jedem dieser 6 Beispiele eine der Anforderungen erfüllt ist, aber die anderen nicht. [3 P.]

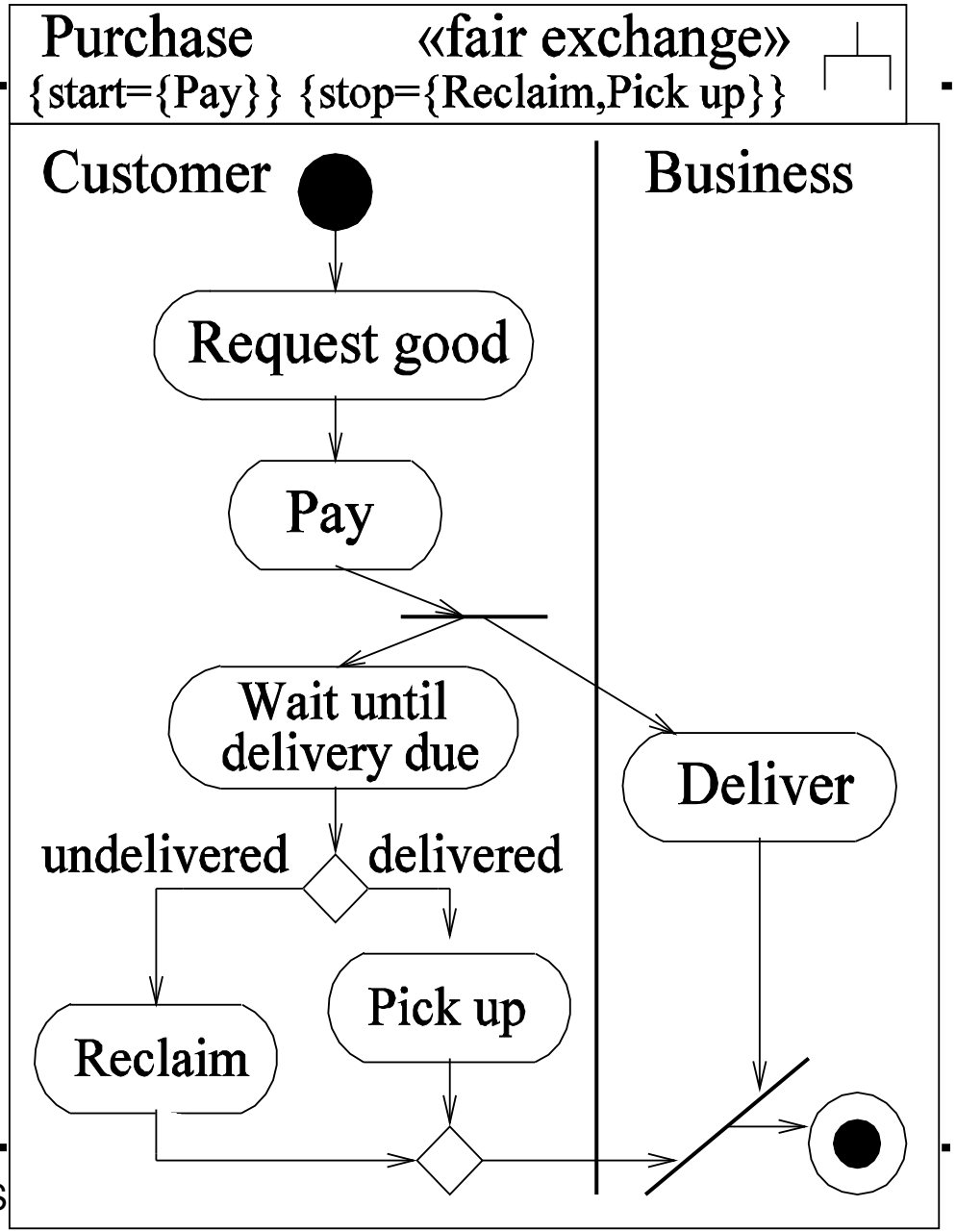
b) Gebe zwei der genannten Sicherheits-anforderungen an, die sich gegenseitig ausschliessen. [1 P.]

Sichere e-Transaktionen

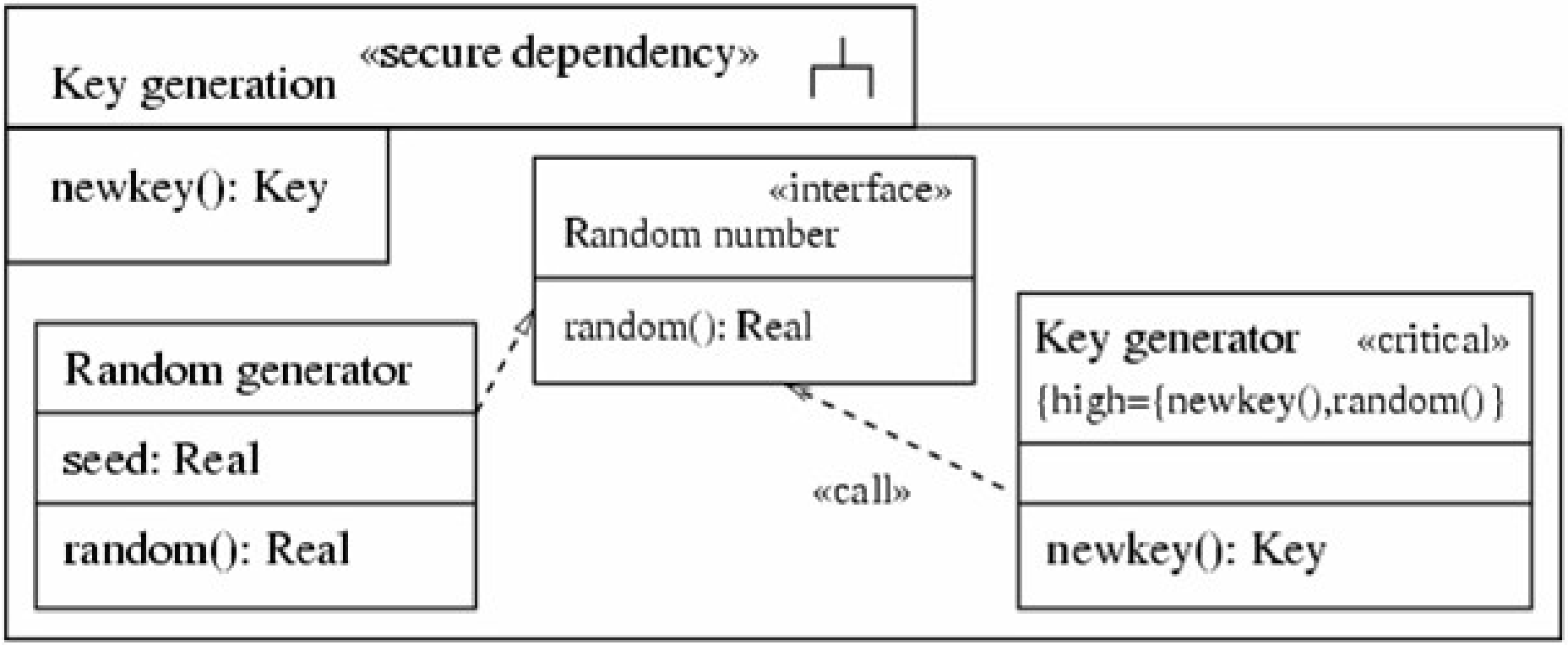
Sicherheit von Geschäftsprozessen z.B. bei e-Transaktionen.

Hier: Kunde kauft Ware beim Händler.

Nach Bezahlung bekommt Kunde Ware ausgeliefert oder kann Bezahlung zurückfordern.

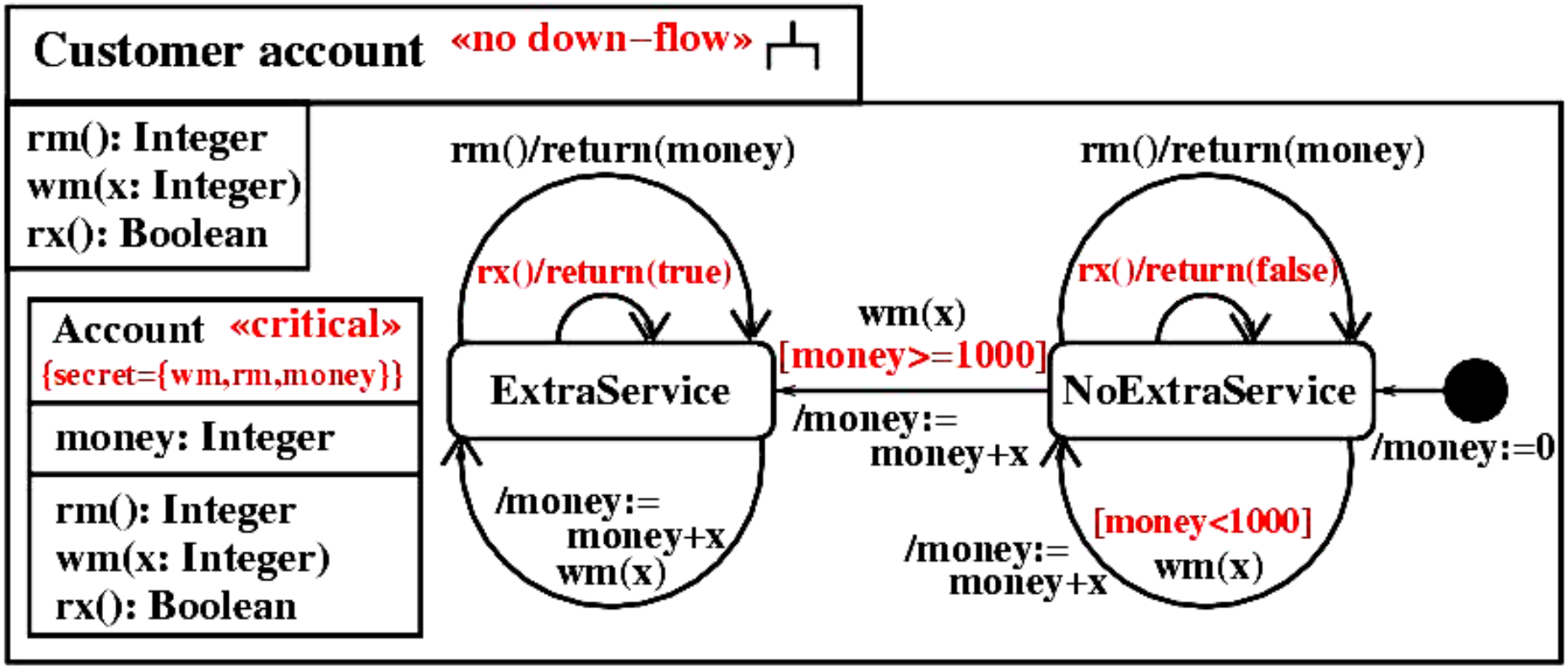


Durchgängige Datensicherheit



Sicherheitslevel definieren. Konsistenzanalyse.

Versteckte Informationsflüsse



Können vertrauliche Daten heraussickern ? Oft ohne Werkzeugunterstützung nicht ersichtlich.

Sicherheitsanforderungen

Jeweils

- verschiedene Sicherheitsstufen (Gewichtungen) bzgl. einzelner Daten
- Berücksichtigung verschiedener möglicher Angreifer

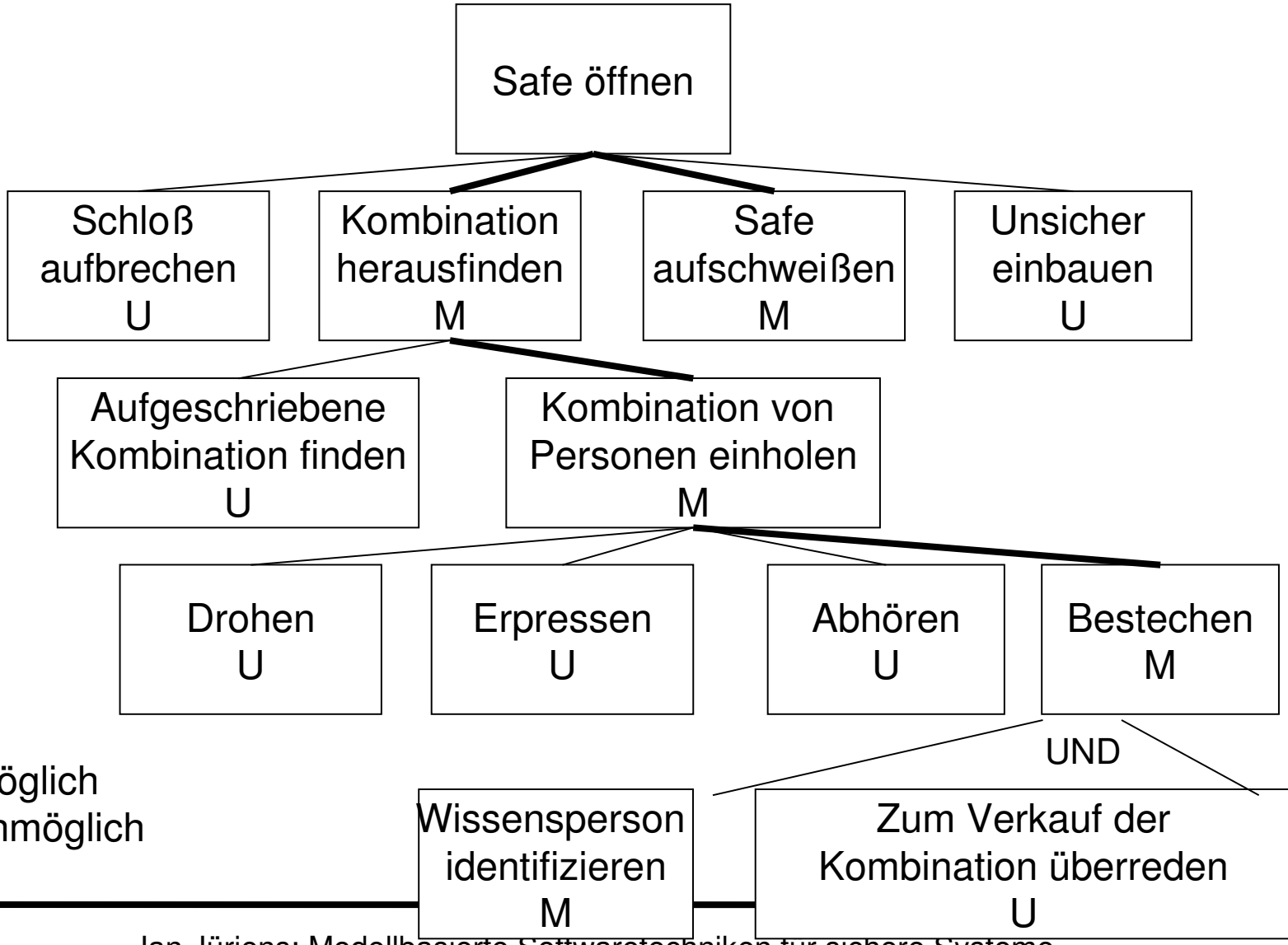
Aufgabe 2

a) Wie unterscheiden sich die o.g. IT-Datensicherheitsanforderungen von System-Sicherheitsanforderungen an verlässliche Systeme in Domänen wie Avionik, Automobil etc. ? [2 P.]

b) Warum ist Datensicherheit in der digitalen Welt schwieriger zu erreichen, als in der physikalischen ? [2 P.]

Angriffsbäume

[Weck 2003]



M = Möglich
U = Unmöglich

Aufgabe 3

- Zeichne einen Angriffsbaum für die IT-Sicherheitsrisiken beim Internet-Banking (ohne Kosten). [6 P.]