Freie Universität Berlin

Security Engineering

Einführung und Überblick

Martin Gruhn
Institut für Informatik
FU Berlin
17. Januar 2008

Inhalt

- Begriffe und Definitionen
  - Was ist IT-Sicherheit?
  - Was ist Security Engineering?
- Security Engineering
  - Prinzipien
  - Sicherheit im Entwicklungsprozess
  - Mechanismen, Maßnahmen, Werkzeuge
  - Evaluierung und Zertifizierung
- Aktuelle (Forschungs-) Aktivitäten

Begriffe und Definitionen

Freie Universität Berlin

- Informationssicherheit (security) nach [Eckert04]

Schutz von IT-Systemen vor unautorisiertem Informationsgewinn und Veränderung von gespeicherten oder verarbeiteten Informationen.

- Elementare Schutzziele:
  - Vertraulichkeit, Unversehrtheit, Authentizität
  - Außerdem: Verbindlichkeit, Verfügbarkeit, Anonymität, ...

- IT-System umfasst:
  - Systemsicherheit und Netzsicherheit (schwer trennbar)
  - Hardware und Software (im Internet meist nur SW)

- Funktionssicherheit (safety):
  - Funktionalität erfüllt Spezifikation
  - Informationssicherheit/Schutz ist Spezialisierung von Funktionssicherheit
- Weitere Nomenklatur:
  - Schutz (security) vs. Sicherheit (safety)

- Datenschutz (privacy):
  - Selbstbestimmung über persönliche Informationen

Was ist Security Engineering?

Freie Universität Berlin

Security Engineering behandelt Werkzeuge, Prozesse und Methoden für Entwurf, Implementierung und Test von sicheren IT-Systemen.

R. Anderson

Sicherheitsbewusste Softwaretechnik

P. Löhr

- Betont ganzheitlichen Ansatz im Gegensatz zu einem speziellen anzuwendenden Schutzverfahren
- Bezeichnungen für LVs, z. B.:
  - IT-Sicherheit (FU Berlin)
  - Security Engineering (TU Darmstadt, Uni Mannheim)

Freie Universität Berlin

Standardwerk von Ross Anderson: “Security Engineering”

- Protokolle
- Zugriffsschutz
- Kryptographie
- Mehrschichtige Sicherheit
- Mehrseitige Sicherheit
- Überwachung
- Biometrie
- Physischer Verfälschungsschutz
- Kopierschutz und Datenschutz
- Prozesse, Management und Evaluierung
- … und viele Szenarien und Beispiele

Freie Universität Berlin

Standardwerk von Ross Anderson: “Security Engineering”

- Protokolle
- Zugriffsschutz
- Kryptographie
- Mehrschichtige Sicherheit
- Mehrseitige Sicherheit
- Überwachung
- Biometrie
- Physischer Verfälschungsschutz
- Kopierschutz und Datenschutz
- **Prozesse, Management und Evaluierung**
- … und viele Szenarien und Beispiele

Probleme bei IT-Sicherheit

Freie Universität Berlin

- Grundsatzproblem
  - Software Engineering *stellt sicher*, dass die *Spezifikation unter vorgesehenen Bedingungen* erfüllt wird.
  - Security Engineering soll *verhindern*, dass ein Schutzziel insb. unter unvorhergesehenen Bedingungen verletzt wird (-> Angreifer).
- Außerdem (z.B. nach [DevStu00])
  - Entwickler sind keine Sicherheitsfachleute
  - Kunde: Funktionalität vor Sicherheit
  - Unabhängige Behandlung von funktionalen und sicherheitsrelevanten Anforderungen
  - Sicherheit häufig am Ende des Entwicklungsprozesses

Inhalt

- Begriffe und Definitionen
  - Was ist IT-Sicherheit?
  - Was ist Security Engineering?
- Security Engineering
  - Prinzipien
  - Sicherheit im Entwicklungsprozess
  - Mechanismen, Maßnahmen, Werkzeuge
  - Evaluierung und Zertifizierung
- Aktuelle (Forschungs-) Aktivitäten

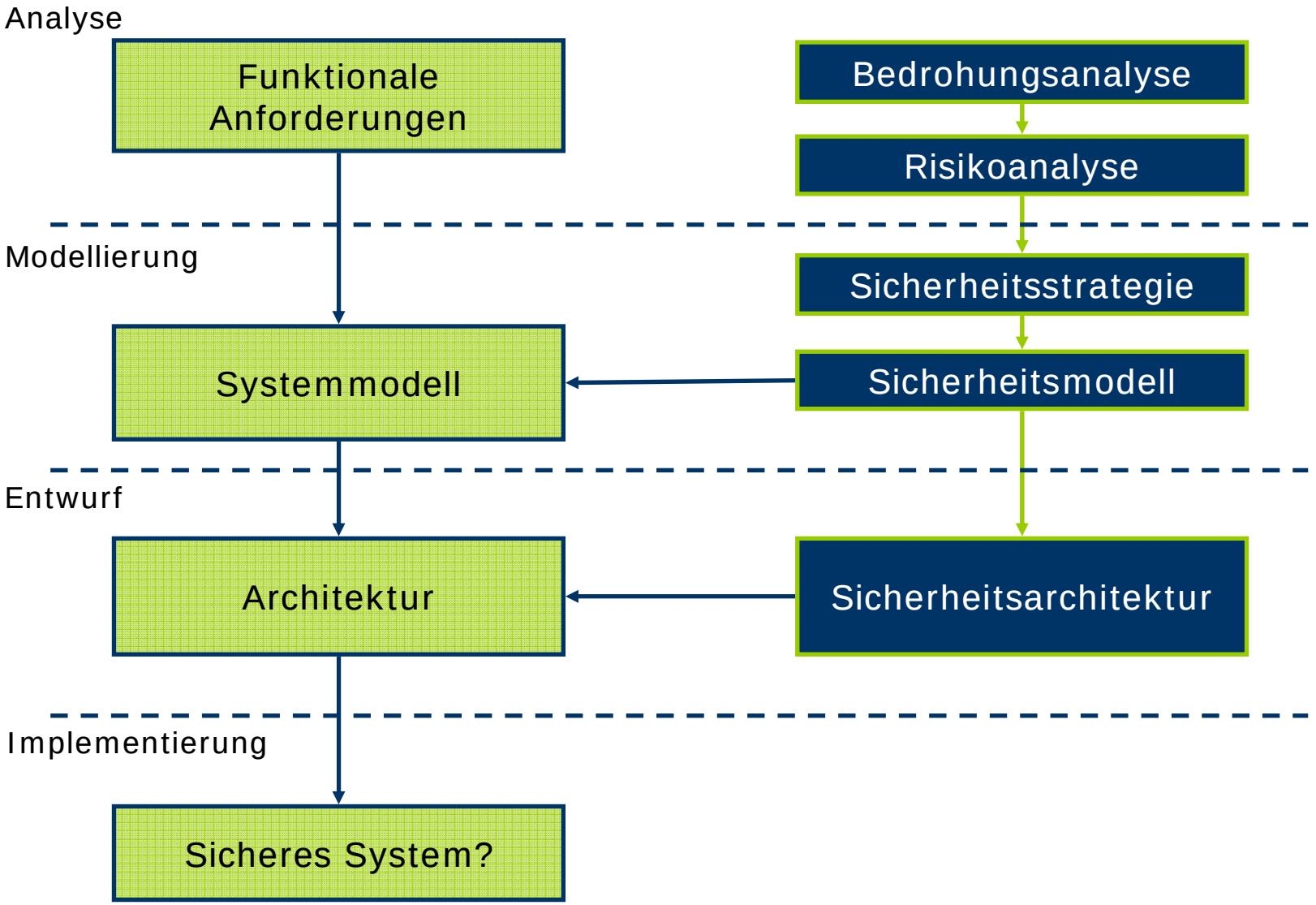
Prinzipien für sichere IT-Systeme

8 Prinzipien nach Salzer/Schröder (SalSch75)

- Economy of mechanism
  - „Keep the design as simple and small as possible.“
  - Begünstigt Durchsichten und Debugging
- Fail-safe defaults
  - „The default situation is lack of access.“
  - Lückenhafte Konfiguration tendiert zu sicheren Zuständen
- Complete mediation
  - „Every access to every object must be checked for authority.“
  - Vorsicht vor Optimierung Zugriffsschutzmechanismen
- Open design
  - „The design should not be secret“

- Separation of privilege
  - „Two keys“
  - Vier-Augen-Prinzip
- Least privilege
  - „Need-to-know“
  - So wenig Rechte wie möglich zur Aufgabenbewältigung
- Least common mechanism
  - Minimieren potentieller Informationsflüsse zwischen Nutzern -> Möglichst auf einen Nutzer beschränkte Prozesse
- Psychological acceptability
  - „Mental image of his protection goals matches the mechanisms to use“

IT-Sicherheit im Entwicklungsprozess



Grundlage bildet BSI Grundschutzhandbuch:

1. Analyse:
   - Strukturanalyse:
     - Funktionale Anforderungen, Einsatzumgebung (Topologieplan); Funktionale Komponenten
   - Schutzbedarfsermittlung:
     - Mit Hilfe von Schadenszenarien (Auswirkungen auf Aufgabenerfüllung, Unversehrtheit, finanziell)
   - Bedrohungsanalyse:
     - z. B. mit Bedrohungsbaum, Bedrohungsmatrix
   - Risikoanalyse:
     - Kombination von Wahrscheinlichkeit und potentiellem Schaden eines Angriffs

2. Modellierung:
   - Sicherheitsstrategie:
     - Bestimmung der Maßnahmen zur Erfüllung des Schutzbedarfs (z.B. Authentisierung, Rechteverwaltung)
   - Sicherheitsmodell:
     - Rollenmodell (Rollen-Rechte-Ressourcen); evtl. formal
3. Sicherheitsarchitektur:
   - Konkrete Umsetzung der Sicherheitsstrategie in detaillierte Architektur
4. Umsetzung
   - Implementierung
   - Validierung: Modul- und Integrationstests; Code-Inspektionen; Penetrationstests
5. Aufrechterhaltung im laufenden Betrieb

- Kritik an Entwicklungsprozess für sichere SW
  - Ähnelt Wasserfallmodell
  - Was ist mit agilen Ansätzen?
  - Was ist mit open source Projekten?

- Integration in Softwareentwicklungsprozess?

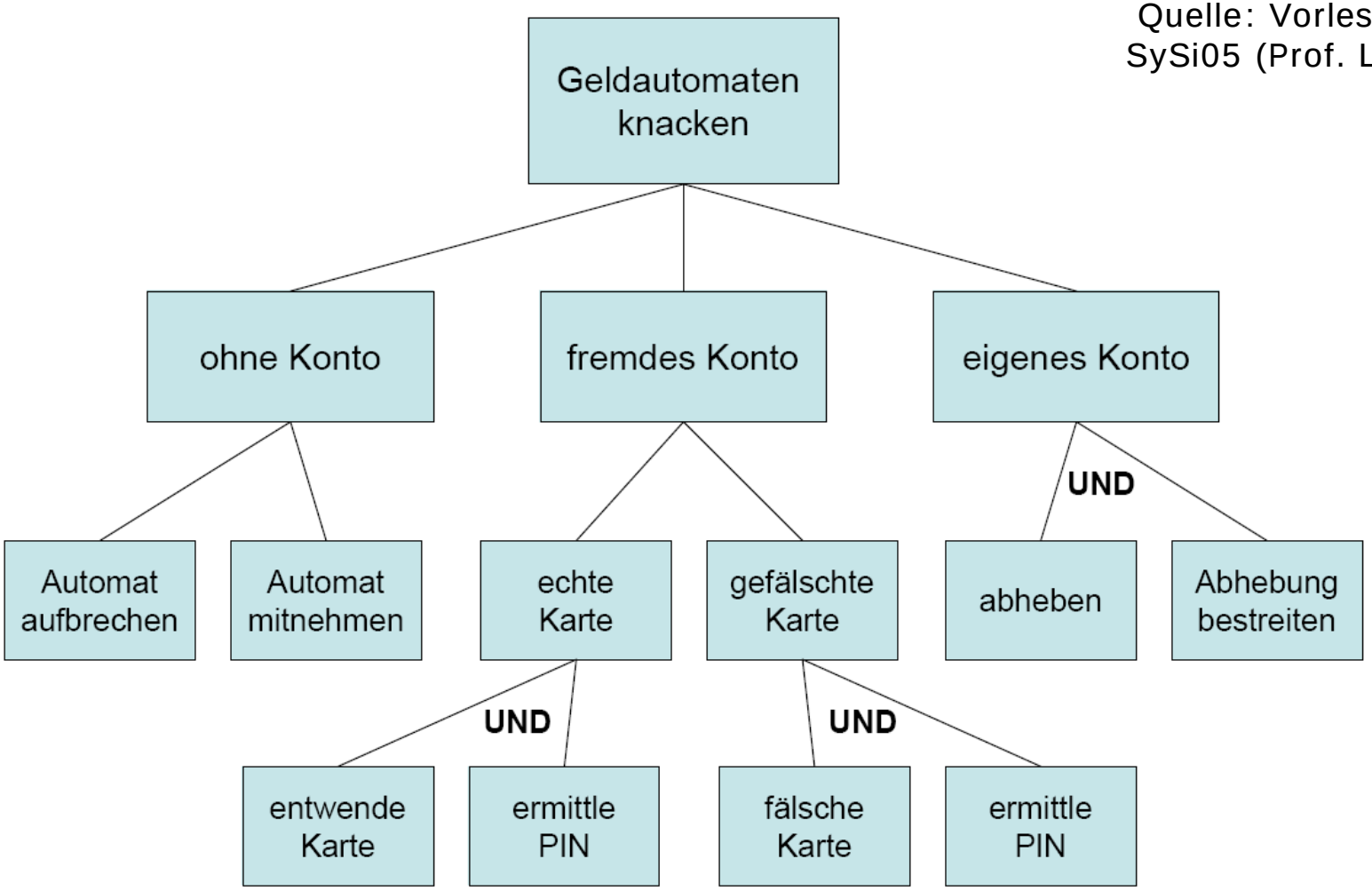
Maßnahmen zur Analyse

Freie Universität Berlin

- Bedrohungsbaum
  - Wurzel: Bestimmte Bedrohung
  - Knoten: Teilziel
  - Und/Oder Verknüpfungen
  - Angriffsvariante: Ein Weg von Wurzel zum Blatt
  - Textuell oder graphisch (Beispiel folgt)
- Bedrohungsmatrix
  - Klärt: Welches Angriffsszenario kann durch wen (oder was) in welchem Bereich ausgeführt werden
  - Gefährdungsbereiche (Zeilen)
  - Bedrohungsauslöser (Spalten)
  - Angriffsszenarien (Felder)

- Bedrohungsbaum

Quelle: Vorlesung SySi05 (Prof. Löhr)



- Bedrohungsmatrix

Quelle: Vorlesung SySi05 (Prof. Löhr)

	Programmierer	interner Benutzer	externer Benutzer	mobiler Code
HW-Angriffe	-	Sicherungskopie vernichten	Einbruch ...	-
SW-Angriffe	Salami-Taktik	Dateischutz umgehen	Passwort knacken	Vireninfektion
Ressourcen-Blockade	Speicherverbrauch	Prozesse erzeugen	Netzlast erzeugen	Prozessor monopolisieren

Werkzeuge: Entwurf

Freie Universität Berlin

Modellierung formaler Sicherheitsstrategie
SecTOOL [KolKocLoh06]:

- Für UML-basierte Softwareentwicklung
- Hilft bei Erstellung formaler Sicherheitsstrategie bzgl. Zugriffsschutz als VPL (View Policy Language)
- Nutzung von:
  - Use-Case diagram (identifiziert Rollen)
  - Klassendiagramm (identifiziert Objektschnittstellen)
  - Sequenzdiagramme (identifiziert Zugriffsrechte)
- Visualisierung der VPL als "view diagram“
  - Verspricht Round-Trip engineering

Werkzeuge: Analyse, Test

Freie Universität Berlin

- Statische Codeanalyse: z. B. Lapse [LivLam05]
  - Hilft bei Code-Durchsicht
  - Eclipse Plugin für automatisierte statische Analyse von J2EE Webanwendungen
  - Prüft auf: Parameter-Manipulation, SQL injection, Header Manipulierung, ...
  - OSS unter GPL
- Fuzzing
  - Penetrationstests mit automatisch generierten verschiedenartigen Anfragen
  - Erstes Tool: Generieren von Zufallswerten zum Testen von Unix-Kommandos
  - Spezielle Werkzeuge für verschiedene Gebiete, z. B. FTP, ActiveX-Controls, ...

Werkzeuge: Aufrechterhaltung

Freie Universität Berlin

- Der elektronische Sicherheitsinspektor eSI
  - Produkt vom FhI SIT (Sichere Informations-Technologie)
  - Ziel: Test und Überwachung eines laufenden Systems
  - Architektur ermöglicht automatisierte Anwendung existierender Prüfwerkzeuge und Analysetools, z.B. Portscanner, Virenscanner, etc.

Inhalt

- Begriffe und Definitionen
  - Was ist IT-Sicherheit?
  - Was ist Security Engineering?
- Security Engineering
  - Prinzipien
  - Sicherheit im Entwicklungsprozess
  - Mechanismen, Maßnahmen, Werkzeuge
  - **Evaluierung und Zertifizierung**
- Aktuelle (Forschungs-) Aktivitäten

Evaluierung: Produktorientiert

Freie Universität Berlin

- Ziel:
  - Vertrauen in Sicherheit von unabh. zertifizierten Systemen
- TCSEC: 1985 vom DOD, „Orange Book“
  - Bewertung anhand erbrachter Funktionalität
  - 7 Stufen von D-A, z.B.
    - C1/C2: Benutzerbestimmbarer Schutz (Klassischer Zugriffschutz)
    - B1/B2/B3: Systembestimmter Schutz (Informationsflusskontrolle)
  - Kritik: Keine Bewertung der Wirksamkeit von Funktionalität
- ITSEC: 1991, europäischer Standard
  - Funktionsklassen angelehnt an TCSEC
  - Neu: Qualitätsklassen
  - Kritik: Keine Anwendersicht

- Common Criteria (CC), 1999, international
  - Homogenisierung und Erweiterung nationaler Standards
  - Neu: Anwendersicht -> Schutzprofile
- Schutzprofil (Protection Profile, PP)
  - Enthält Sicherheitsanforderungen, Implementierungs- und Produktunabhängig
  - Security Target (ST): Spezialisierung auf konkretes Produkt
  - ST und PP sind selbst evaluierbar
- Ziel:
  - Wachsender Katalog von PPs und evaluierten Produkten

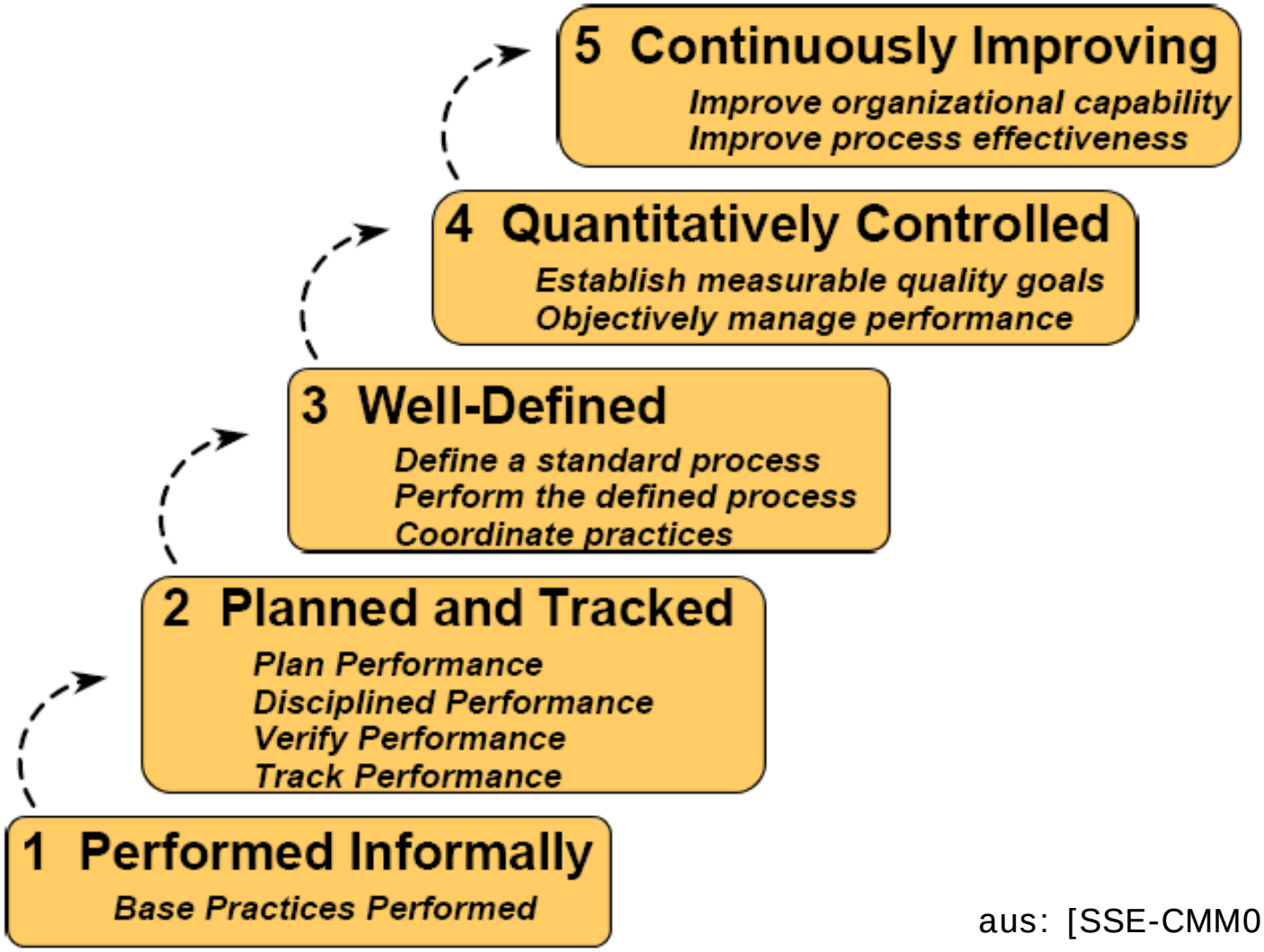
- Nutzen von CC:
  - PPs enthalten Bedrohungen und entsprechende Schutzziele (funktional wie qualitativ) zur Vermeidung
  - Kann als Checkliste dienen
- Kritik an CC (nach Anderson):
  - Nimmt Wasserfallmodell an: Auswirkungen von Änderungen im Produkt sind schwierig zu betrachten
  - Großer Aufwand, hohe Kosten, Bürokratie
  - Zielt auf hierarchisch verwaltete, zentrale Systeme
  - Dokumente sind für normale Entwickler schwer verständlich
  - Was ist mit realer Anwendungsnähe?
  - Was ist mit Implementierungsfehlern und Anwendungsfehlern?

Evaluierung: Prozessorientiert

Systems Security Engineering – Capability Maturity Model (SSE-CMM)

- Reifegradmodell zur Beurteilung der Qualität von Prozessen bei Entwicklung sicherer IT-Systeme
- Definiert:
  - Prozesse (processes) für definierte Gebiete (process area)
  - Goals -> Practices
  - Einstufung mit Capability Measure (Level 1-5)

SSE-CMM: Capability Measure



aus: [SSE-CMM03]

SSE-CMM Process Areas

Freie Universität Berlin

- Organizational process areas:
  - Define Organization‘s Security Engineering Process
  - Improve Organization‘s Security Engineering Process
  - ...
- Project process areas:
  - Ensure Quality
  - Manage Configurations
  - Manage Program Risk
  - ...
- Engineering process areas
  - nächste Folie

SSE-CMM Engineering PAs

Freie Universität Berlin

- Risk area
  - PA 04: Assess Threat
  - PA 05: Assess Vulnerability
  - PA 02: Assess Impact
  - PA 03: Assess Security Risk
- Engineering area
  - PA 10: Specify Security Needs
  - PA 09: Provide Security Input
  - PA 07: Coordinate Security
  - PA 01: Administer Security Controls
  - PA 08: Monitor Security Posture
- Assurance area
  - PA 11: Verify and Validate Security
  - PA 06: Build Assurance Argument

SSE-CMM Engineering PAs

Freie Universität Berlin

- Risk area
  - PA 04: Assess Threat
    - Goal: Threats to the security are identified and characterized
      - BP 04.01: Identify Natural Threats
      - BP 04.02: Identify Man-made Threats
      - ...

- Fazit SSE-CMM:
  - Umfangreiches Modell
  - Nutzung als Kriterienkatalog für Beurteilung von Prozessen für IT-Sicherheit

Inhalt

- Begriffe und Definitionen
  - Was ist IT-Sicherheit?
  - Was ist Security Engineering?
- Security Engineering
  - Prinzipien
  - Sicherheit im Entwicklungsprozess
  - Mechanismen, Maßnahmen
  - Werkzeuge
  - Evaluierung und Zertifizierung
- Aktuelle (Forschungs-) Aktivitäten

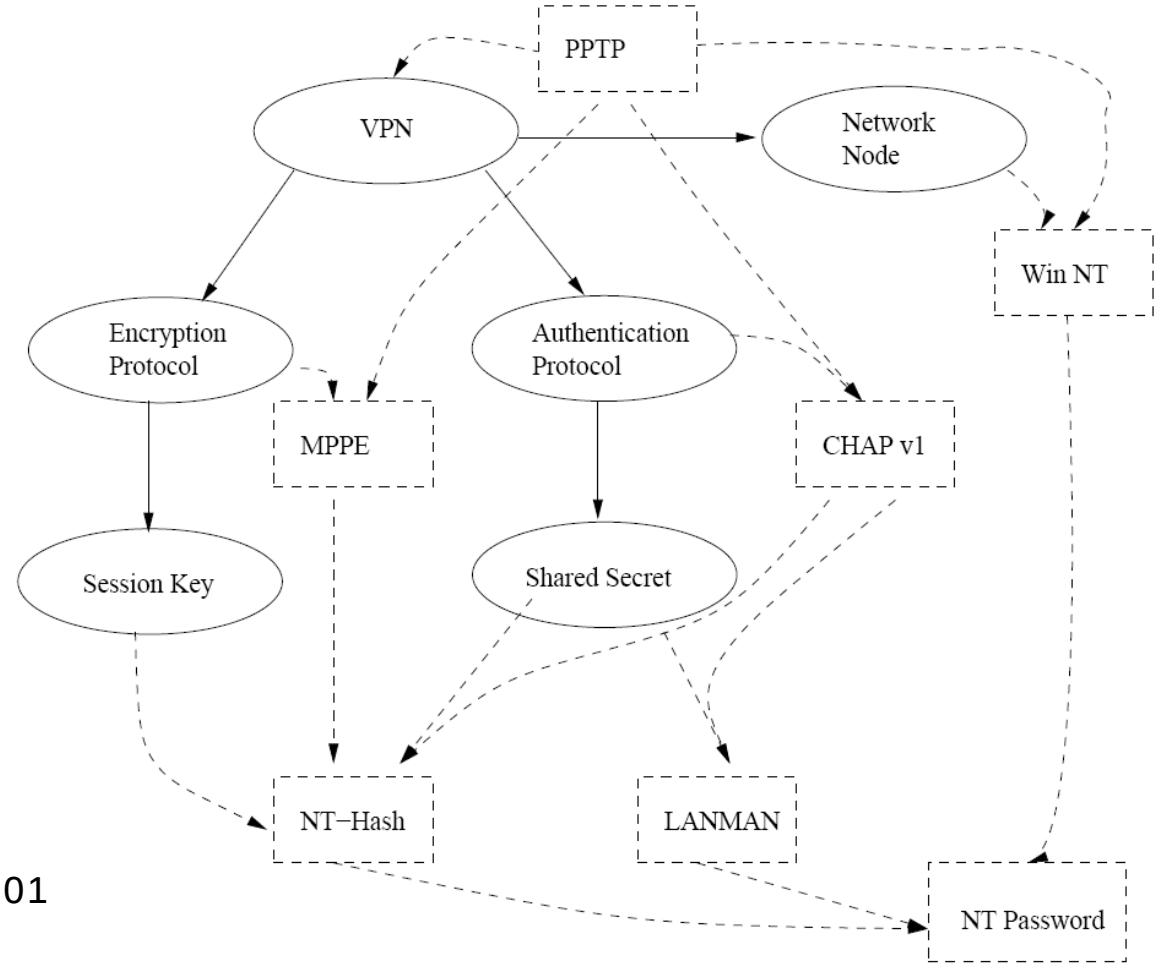
(Forschungs-) Aktivitäten

Freie Universität Berlin

- Security Requirements Engineering und XP [BosWayBod06]
  - Erweitern des Planning Game: *Abuser stories; Security-related user stories*

- Security Patterns [Schumacher03]
  - In der Tradition von Entwurfsmustern von GoF (Gamma, Helm, Johnson, Vlissides)
  - Muster besteht aus: Name, Kontext, Problem, Lösung

- Security Pattern Systems (M. Schumacher)

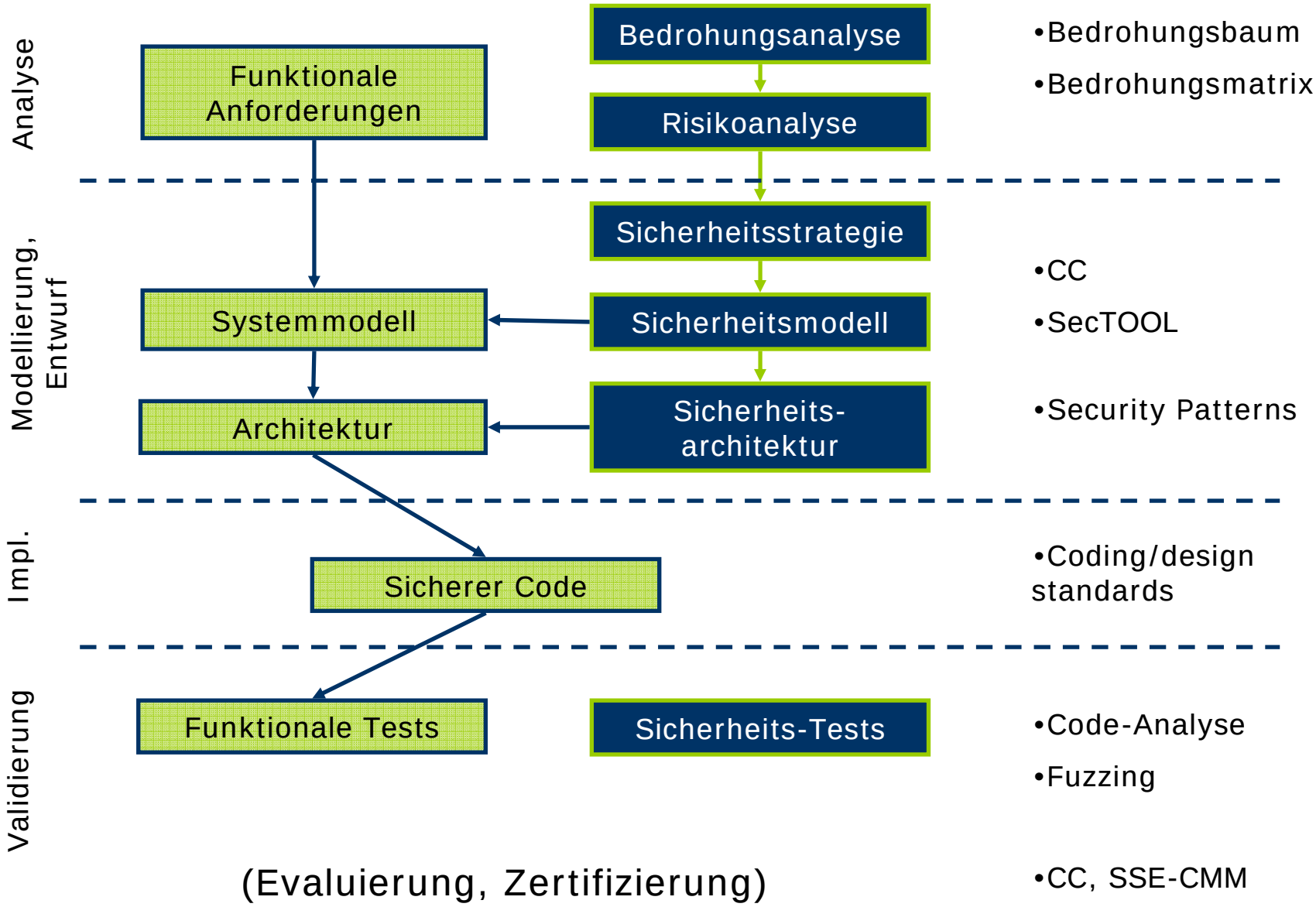


aus: Schumacher01

- Programmier-Ratgeber
  - Viega/McGraw (2001): Building Secure Software
  - Security Engineering: Patterns & Practices (Microsoft)
  - Tech Tips der CERT (Programm des SEI der Carnegie Mellon University) http://www.cert.org/tech_tips/
  - Ähnlich: Ranglisten häufigster Fehler in Webanwendungen und deren Vermeidung (z.B. [OWASP07])
- Security Engineering und Open Source Software
  - Verstehen von Prozessen für sichere OSS -> Empfehlungen
  - Master-Arbeit
    - Analyse, Beschreibung und Bewertung sicherheitsrelevanten Verhaltens im Rahmen der Entwicklung von OSS
    - Qual. Patch-Tracker Analyse bzgl. Sicherheitspatches
    - Was gibt es für sicherheitskritische Fehler, wie sind sie entstanden und wie wird damit umgegangen?

Zusammenfassung: Sicherheitsprozess

Freie Universität Berlin



Quellen

[Anderson01] R. Anderson (2001): Security Engineering.

[BosWayBod06] G. Boström et al. (2006): Extending XP Practices to Support Security Requirements Engineering.

[Eckert04] C. Eckert (2004): IT-Sicherheit.

[eSI05] eSI- Der elektronischer Sicherheitsinspektor. http://www.sit.fraunhofer.de/fhg/Images/eSI_de_en_tcm105-96939.pdf

[KolKocLoh06] Kolarczyk et. al (2006): SecTOOL -- Supporting Requirements Engineering for Access Control

[LivLam05] B. Livschits, M. Lam (2005): Finding Security Errors in Java Programs with Static Analysis.

[OWASP07] OWASP Top 10 2007. http://www.owasp.org/index.php/Top_10_2007

[SalSch75] J. Saltzer, D. Schroeder (1975): The Protection of Information in Computer Systems.

[Schumacher03] M. Schumacher (2003): Security Engineering with Patterns.

[SSE-CMM03] Systems Security Engineering Capability Maturity Model 3.0. http://www.sse-cmm.org/

Vielen Dank!