Goldene Regeln der IT-Sicherheit bei der Beauftragung und Erstellung von Software

Dr. Boris Hemkemeier, Commerzbank
Patrick Hildenbrand, SAP
Tom Schröer, SAP

OWASP
Frankfurt, 25.11.08


Einleitung und Motivation
10 Goldene Regeln für Entwickler
Rollenspezifische Regeln
10 Goldene Regeln für Auftraggeber

- Dieser Vortrag und das dazugehörige White-Paper wurden von der Commerzbank AG und der SAP AG im Rahmen des Secologic Forschungsprojektes erstellt, (Laufzeit 2005 - 2007), nähere Informationen unter www.secologic.de.

Secologic Projektkonsortium

SAP AG	Commerzbank AG
Eurosec GmbH	TU Hamburg

Die Autoren des Whitepapers sind

Dr. Kai Buchholz-Stepputtis (Commerzbank)	Tom Schröer (SAP)
Dr. Boris Hemkemeier (Commerzbank)	Rosemaria Giesecke (SAP)

- Wir bedanken uns beim Bundesministerium für Wirtschaft und Technologie für die Förderung dieses Projektes.

- Die Verwendung der Inhalte dieser Folien ist unter Verweis auf die Urheber ausdrücklich erwünscht.

Übersicht Projektergebnisse Secologic

- Ergebnisse zu Themen rund um das Thema Anwendungssicherheit.
  - Unterschiedliche Detailstufen
    - Überblick
    - Anleitungen
    - Schulungsunterlagen
    - Implementierung einer Reihe prototypischer Lösungen
    - Lernanwendung SecologicTrain
  - Unterschiedliche Darstellungsformen
    - Wissenschaftliche Veröffentlichungen
    - Pragmatische Lösungsansätze
  - Vergleich von verschiedenen Lösung eines Geschäftsszenarios im Web-Services

- Ergebnisstruktur
  - Sicherheit im Softwarelebenszyklus

    Goldene Regeln und organisatorische Maßnahmen
  - Sicherheit in Applikationen
    - Programmiersprachen

      PHP, Java, C/C++, Javascript, Web-Services
    - Angriffe und Lösungen

      Session Management, XSS, SQL Injection, Cross Site Request Forgery/Session Riding/Session Hijacking
  - Sicherheitstests
    - Vergleich von Testwerkzeugen
    - Penetrationstests, Testen von Web Services

Warum „10 Goldene Regeln“ (allgemein) ?

- Allgemeine Zielstellung „Goldener Regeln“:
  - „Goldene Regeln“ („Golden Rules“) sollen kurz und verständlich auf die wichtigsten Kernpunkte hinweisen
  - Zahl „10“ als guter, gerade noch „verdaulicher“ Ansatz
  - Beginn mit solchen „Best Practices“ kann manchmal mehr bewirken
    als mehrere „Festmeter“ an Dokumenten
  - Nach erster Sensibilisierung und „Awareness“ sind weitere Vertiefungen und Konkretisierungen immer noch möglich
  - Oft ist eine solche erste Sensibilisierung mit handlichen kurzen Regeln
    sogar der beste erste Schritt
  - Das gilt je nach Zielgruppe auch für IT-Sicherheit

„10 Goldene Regeln“ auch für IT-Sicherheit ?

- IT-Sicherheit wird oft schwarz/weiß (unsicher/sicher) betrachtet, das trifft in der Praxis aber in der Regel nicht zu
- Auch bei IT-Sicherheit sind „Best Practices“ und „Goldene Regeln“ als erster Ansatz aber sinnvoll
- Ohnehin unterliegen nicht alle Anwendungen immer den allerhöchsten Sicherheitsanforderungen

- „Best Practices“ machen Sinn auch für „Hochsicherheits-Produzenten“
  wie renommierte Banken oder Softwarehersteller ;-)
- Hier folgt dann immer auch Vertiefung und Detailarbeit, dafür gibt es dann aber entsprechende Spezialisten
- Aber auch hier gibt es Klientel, denen die Thematik am besten zunächst mit verständlichen Regeln nähergebracht werden kann

„10 Goldene Regeln der IT-Sicherheit“

- Inhaltlicher Schwerpunkt des Secologic Projektes: Entwicklung sicherer Software
- Erster Ansatz im Secologic Projekt: „10 Goldene Regeln der IT-Sicherheit“ für Entwickler
- Ausgangsbasis:
  - Verschiedene „Top Ten“ Ansätze für „Software Coding“
  - Ansätze z.T. heterogen, überwiegend auch englischsprachig
  - Z.T. eher an „Top Ten Vulnerabilities“ orientiert oder zu umfangreich (lesenswert hier aber auf alle Fälle „OWASP Top Ten“ und „OWASP Guide“)
- Erste Zielstellung dazu im Secologic Projekt:
  - Deutschsprachige Konsolidierung
  - Schwerpunkt auf lösungsorientierte „Goldene Regeln“ als „Best Practices“


Einleitung und Motivation
10 Goldene Regeln für Entwickler
Rollenspezifische Regeln
10 Goldene Regeln für Auftraggeber

Misstrauen Sie jeder (Benutzer-) Eingabe

- Jede Eingabe kann unerwartet sein. Eingaben können nicht nur vom GUI kommen – sondern auch aus Dateien und Serviceaufrufen. Ein blindes Vertrauen ist heute die häufigste Sicherheitsschwachstelle in Applikationen.
- Ablauf für die Prüfung
  - Alles in eine kanonische Form bringen
  - Bekannte Angriffe filtern (Blacklist) (wichtiger Schritt, aber nicht ausreichend!)
  - Nur erwartete Eingaben annehmen (Whitelist)
- Grundregeln
  - Alle Eingaben prüfen
  - Alle Steuerzeichen erkennen
  - Am Server prüfen

„Minimale Rechte-Prinzip“

- Zu viele Rechte sind ein Sicherheitsrisiko
  - Benutzern stets nur die Berechtigungen geben, die sie zur Erfüllung ihrer Aufgabe wirklich brauchen
  - Berechtigungen nicht hart codieren, sondern konfigurierbar machen
  - Gibt es unterschiedliche Adminstrationsaufgaben?
  - „Minimale Rechte-Prinzip“ gilt auch für:
    - Technische Benutzer
    - Betriebssystem- /Datenbankbenutzer
    - Zugriffsberechtigungen im Dateisystem
  - Vorsicht auch, wenn auf Daten oder Funktionen über mehrere Wege zugegriffen werden kann

Keine unnötige Veröffentlichung von internen Daten!

- Achten Sie darauf, wer welche Daten zu sehen bekommt.
- Auch Stellen außerhalb der Anwendungsfunktionen berücksichtigen wie :
  - Fehlermeldungen mit zu vielen Systeminternas
  - Cookies und URLs
  - standardmäßig aktivierte Debug-Funktionen
  - Systeminformationen, Patchstand
  - Logdateien

Nutzen Sie die Erfahrung von anderen (do's und dont's)

- Andere Entwicklungsprojekte haben schon eine ganze Reihe Erfahrungen gemacht.
- Viele dieser Erfahrungen sind schon von Teamkollegen oder andern Mitarbeitern gemacht worden.
- Weitere sind käuflich erhältlich oder sogar frei verfügbar.

  → Nutzen Sie diese Erfahrungen und machen Sie nicht die gleichen Fehler, die andere schon gemacht haben.

- Es gilt:
  - Erfinden Sie Schwachstellen nicht neu
  - Entwickeln Sie keine eigenen Sicherheitsfunktionen

Sichere Einstellungen

- Sichere Voreinstellungen
- Alle Benutzerkennungen, Passwörter und Verschlüsselungsschlüssel änderbar!
- Machen Sie es einfach!
- Sicherheitsdokumentation

Niemals „Security by Obscurity“ und niemals „Hintertüren“

- „Security by Obscurity“
  - Vertrauen Sie niemals darauf, dass Ihre Software nicht angreifbar ist nur weil der Quellcode bzw. die Sicherheitsmechanismen unbekannt sind

  → Software muss auch dann noch gut sein, wenn der Quellcode bekannt würde
  - Problem: „obskure“ Lösungen sind häufig auch intern nicht dokumentiert
  - Bsp.: Selbstgeschriebene Verschlüsselungsverfahren
- Keine „Hintertüren“
  - versteckte Funktionalitäten
  - versteckte oder undokumentierte Parameter
  - undokumentierte Aufrufe über Benutzereingaben

Gute Fehlerbehandlung (Fail securely)

- Es lassen sich nicht alle möglichen Situationen einer komplexen Anwendungslandschaft vorhersehen oder testen.
- Darum wird es im Betrieb Fehlersituationen geben. Stellen sie sich auf diese Situationen ein.
- Stellen Sie vor allem sicher, dass die Anwendung im Fehlerfall in einen sicheren Zustand übergeht (oder ihn behält) und nicht "unsicher" wird.

Denken Sie an Nachvollziehbarkeit (Trace, Audit, Logs)

- Protokollieren Sie alles, was in der Anwendung an sicherheitskritischen Ereignissen auftritt. Das sind zum Beispiel:
  - Serverstarts und –stopps
  - wesentliche Konfigurationsänderungen
  - unberechtigte Zugriffsversuche
  - Zugriff auf besonders sensible Daten
  - alle Änderungen in der Benutzer- und Berechtigungsverwaltung
- Denken Sie auch an die Archivierung dieser Daten. Protokolle, die schon nach kurzer Zeit überschrieben werden, können danach nicht mehr nachvollzogen werden.
- Denken Sie daran, dass in Protokollen vertrauliche Daten stehen können und implementieren Sie Zugriffsberechtigungen.

Machen Sie Code-Reviews

- Planen Sie als Entwicklungsschritt Code-Reviews durch einen Kollegen fest ein. Diese Code-Reviews haben 2 wesentliche Vorteile:
  - Es gibt eine 2. Meinung zur vorgeschlagenen Lösung. Ohne Vorprägung durch sehr intensive Vorüberlegungen kann ein „unabhängiger“ Kollegen Sicherheitsschwachstellen häufig einfacher finden.
  - Die Qualität des Codes verbessert sich nach und nach, weil jeder Entwickler weiß, dass:
    - sein Code die Reviews durch andere „überleben“ muss und
    - jeder aus den Fehlern von Kollegen lernt und darum nach und nach nachvollziehbareren Code schreibt.

Rechnen Sie mit Schwachstellen: sehen Sie Patches vor

- Es können auch nach Auslieferung der Software neue Bedrohungsszenarien auftreten.
- Es sollte für den Kunden ein nutzerfreundliches Einspielen eines Patches ermöglicht werden. Zudem sollte der Kunde feststellen können, ob er den notwendigen Patch schon eingespielt hat.

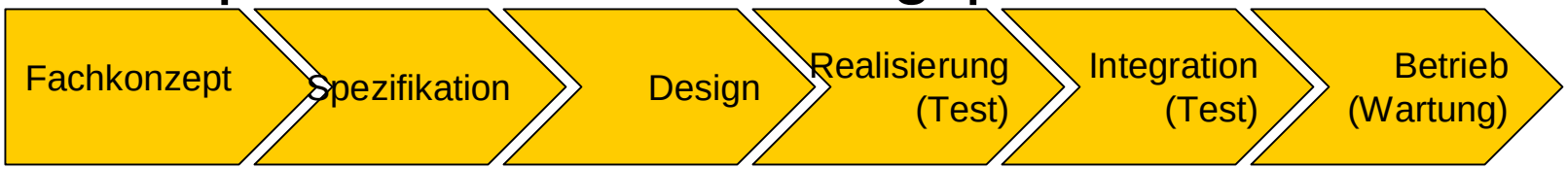

Einleitung und Motivation
10 Goldene Regeln für Entwickler
Rollenspezifische Regeln
10 Goldene Regeln für Auftraggeber

„10 Goldene Regeln der IT-Sicherheit“

- Erster Schritt im Secologic Projekt:
  - „Zehn Goldene Regeln der IT-Sicherheit“ für Entwickler
  - Siehe auch eben gehaltener Präsentationsteil
- Weitere Praxiserfahrung aber:
  - Entwickler handeln nur im Rahmen der Vorgaben einer Projektleitung, die ihrerseits wiederum im Auftrag eines Auftraggebers tätig ist
  - Fazit: Die eigentlichen Hebel für sichere Software liegen woanders
- Außerdem weitere Zielstellung im Secologic Projekt
  - Nutzen möglichst auch für Mittelstand
  - Nutzen möglichst auch dort, wo nicht per se fundiertes Knowhow bzgl. IT-Sicherheit vorausgesetzt werden kann

„10 Goldene Regeln der IT-Sicherheit“ (1 von 4)

- Exemplarischer Entwicklungsprozeß:



- Keine Präjudizierung für bestimmtes Entwicklungsparadigma, Betrachtung hier vielmehr:
  Welche Rollen gibt es typischerweise im Softwareerstellungsprozeß ?
- Nebenthesen und Erfahrungen
  - Je früher Sicherheit im üblichen Erstellungsprozeß integriert ist, desto besser
  - Optimalerweise ist IT-Sicherheit kein “losgelöster”, “isolierter” Prozeß
  - Optimalerweise existieren in allen Teilprozessen geeignete Sicherheitspakete

„10 Goldene Regeln der IT-Sicherheit“ (2 von 4)

- Welche Rollen gibt es immer, unabhängig vom jeweiligen Entwicklungs-Paradigma oder konkreten Erstellungsprozeß:
  - Rolle IT-Entwickler
  - Rolle IT-Projektleiter
  - Rolle Auftraggeber
- Welche Rollen kann es je nach Projektvolumen und Komplexität zusätzlich geben:
  - Rolle IT-Architekt
  - Rolle „Tester“
  - Ggf. weitere Rollen

„10 Goldene Regeln der IT-Sicherheit“ (3 von 4)

- Lösungsansatz:
  Rollenspezifische Ausgestaltung der „Goldenen Regeln der IT-Sicherheit“
- Ausgangsbasis:
  - Im Prinzip kaum Quellen für solche rollenspezifische „Best practices“
  - Nutzen auch für Mittelstand und Non-IT-Sicherheitsexperten denkbar
  - Fazit: Tatsächlicher Mehrwert generierbar
- Vorteil:
  - Ganzheitliche Betrachtung, nicht nur Fokussierung auf „Coding“
  - Jeder benötigt nur die für seine Rolle zugeschnittenen Regeln

„10 Goldene Regeln der IT-Sicherheit“ (4 von 4)

- Gesamtübersicht aller „10 Goldenen Regeln“ für alle Rollen

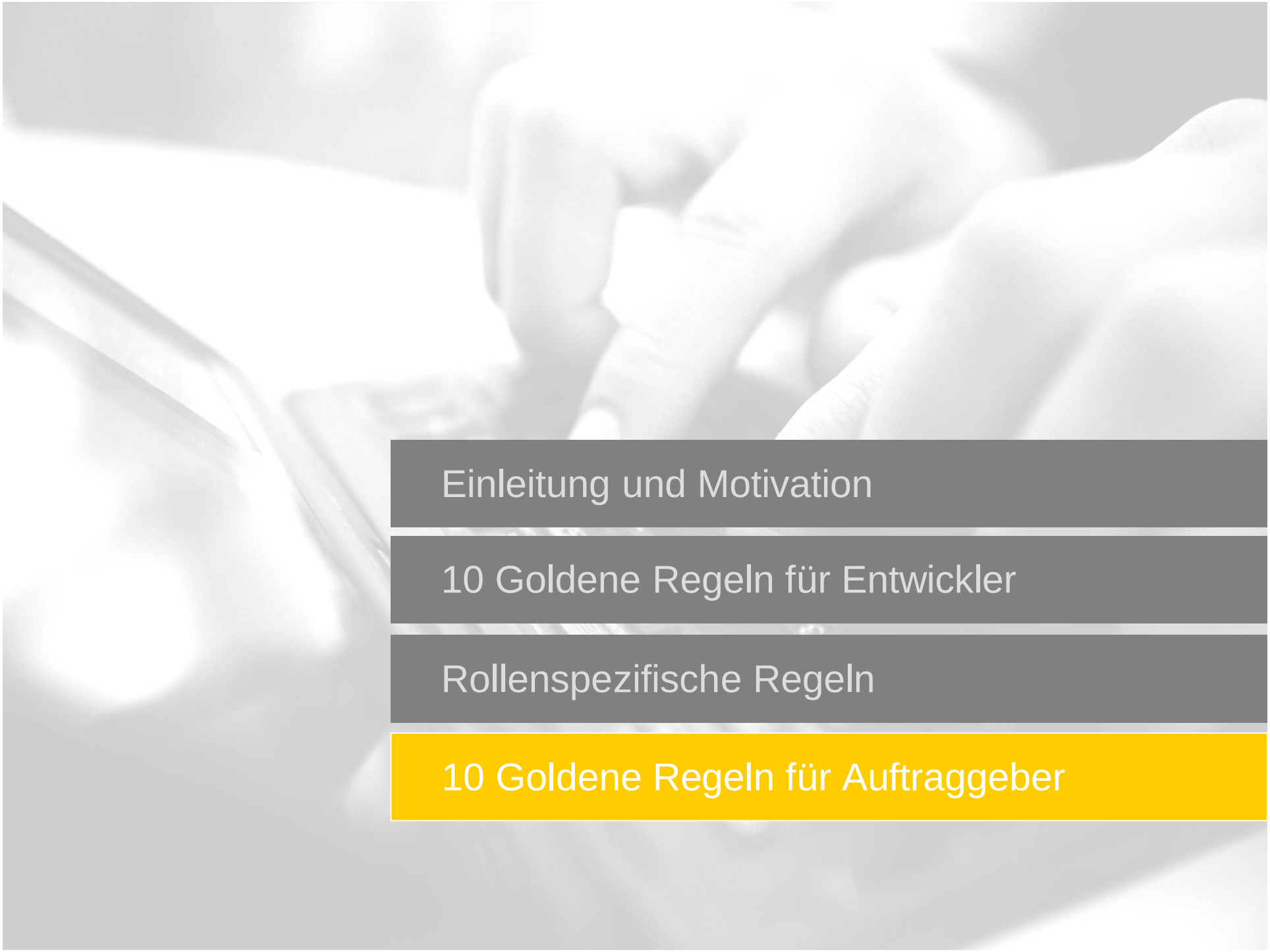
Secologic

Entwicklung sicherer Software: Jeweils *10 Goldene Regeln* für die beteiligten, unterschiedlichen Rollen

Analyst/Auftraggeber	Planer (Architekt)	Entwickler	Tester	Projektleiter
Denken Sie über geschäftliche Konsequenzen von IT-Sicherheitsfragen nach	Nutzen Sie die Erfahrung von anderen: • Erfinden Sie Schwachstellen nicht neu • Entwickeln Sie keine eigenen Sicherheitsfunktionen! • Internationale Sicherheitsstandards (technische Standards)	Misstrauen Sie jeder (Benutzer-) Eingabe (Validate Input)	Testen Sie priorisiert Hauptangriffspunkte	Sie sind verantwortlich für die Umsetzung aller Anforderungen, lesen Sie zunächst also alle anderen "Goldenen Regeln"
Berücksichtigen Sie gesetzliche und regulatorische Anforderungen	„Minimale Rechte"-Prinzip, Berechtigungskonzept erstellen	„Minimale Rechte"-Prinzip	Machen Sie Code-Reviews	Fordern Sie die fachlichen Sicherheitsanforderungen ein
Definieren Sie fachliche Rollen, Zuständigkeiten und Prozesse	Trust Zones: Welchen eingehenden Daten darf vertraut werden und welchen nicht	Keine unnötige Veröffentlichung von internen Daten! (Sanitize Output)	Machen Sie Regressionstests (einmal gefixte Fehler sollen zukünftig nicht mehr auftauchen)	IT-Sicherheit gleich von Beginn an, nicht erst am Ende oder hinterher
Berücksichtigen Sie auch Anforderungen an die Nachvollziehbarkeit	Machen Sie es einfach!	Nutzen Sie die Erfahrung von anderen (do's und dont's) • Erfinden Sie Schwachstellen nicht neu • Entwickeln Sie keine eigenen Sicherheitsfunktionen!	Binden Sie externe Sicherheitsexperten mit ein (für Schwachstellentest und/oder –reviews)	In allen Projektphasen gibt es Arbeitspakete für IT-Sicherheit
Verfügbarkeit und Business Continuity können spezielle Anforderungen sein	Niemals „Security by Obscurity"	Sichere Einstellungen • Sichere Voreinstellungen • Alle Benutzerkennungen, Passwörter und Verschlüsselungsschlüssel änderbar! • Machen Sie es einfach! • Sicherheitsdokumentation	Nutzen Sie vorhandene Testfälle und Tools	Planen Sie hinreichende Ressourcen für IT-Sicherheit
Sicherheitsanforderungen sollten bekannt, vereinbart und fixiert sein	Sichere Voreinstellungen	Niemals „Security by Obscurity" und niemals „Hintertüren"	Testen Sie funktionale Sicherheitsanforderungen	Vergessen Sie nicht Ihre (Projekt-)Mitarbeiter, planen Sie Sensibilisierung und Know-How
Die Erfüllung von Sicherheitsanforderungen sollte abnahmerelevant sein	Machen Sie einen Architekturreview	Gute Fehlerbehandlung (Fail securely)	Testen Sie auf Schwachstellen	Nutzen Sie Standards, sichere Komponenten und existierende Erfahrungen
Vergegenwärtigen Sie sich Restrisiken und treffen Sie ggf. Vorkehrungen	Mehrere Sicherheitshürden (Defense in Depth)	Denken Sie an Nachvollziehbarkeit (Trace, Audit, Logs)	Erstellen Sie einen Testplan	Bewahren Sie Handlungsspielraum für mögliche Veränderungen
Vergessen Sie nicht Ihre eigenen Prozesse, organisatorischen Maßnahmen und Leute	Rechtliche Anforderungen beachten • Datenschutz ermöglichen • Machen Sie alles nachvollziehbar (sofern sinnvoll)	Machen Sie Code-Reviews	Führen Sie Sicherheitstests nicht gleichzeitig mit fachlichen Tests durch	Machen Sie Restrisiken transparent und lassen sich diese abzeichnen
Alles kann sich ändern, seien Sie darauf vorbereitet	Rechnen Sie mit Schwachstellen: sehen Sie Patches vor	Rechnen Sie mit Schwachstellen: sehen Sie Patches vor	Sehen Sie für alle ("normalen") Testfälle Negativbeispiele vor	Denken Sie an diejenigen, die später mit dem Produkt umgehen müssen

Die vorliegende Dokumentation wurde von der Commerzbank und der SAP AG im Rahmen des vom BMWi geförderten Secologic Forschungsprojektes erstellt, (Laufzeit 2005 / 2006), und im Rahmen des *Infodays* am 15.03.2006 in Frankfurt vorgestellt. Nähere Informationen unter: www.secologic.de.

- Ausführliches Detaildokument auf der Secologic-Seite


Einleitung und Motivation
10 Goldene Regeln für Entwickler
Rollenspezifische Regeln
10 Goldene Regeln für Auftraggeber

Regel 1: „Denken Sie an geschäftliche Konsequenzen von IT-Sicherheitsfragen“

- Entscheidender Maßstab sind geschäftliche Anforderungen
  - Wesentliche Sicherheitsanforderungen z.B. Vertraulichkeit und Integrität
  - Typische und geeignete Fragestellungen:
    - „Was wäre wenn ?“
    - „Welcher Schaden könnte eintreten wenn ?“
  - Monetäre Konsequenzen bei Verletzung der Sicherheitsziele, z.B.:
    - Betroffene Informationen, mögliche Szenarien und Bedrohungen
    - Eventuelle Geschädigte, „Nutznießer“ und „Interessenten“
  - Beispiele:
    - „Was wäre wenn“ der Konkurrenz die Einkaufskonditionen bekannt wären ?
    - Ab welcher Zeit kann der Ausfall des Systems unternehmenskritisch werden ?

Regel 2: „Berücksichtigen Sie gesetzliche und regulatorische Anforderungen“

- An Gesetzen und Regularien „kommt man nicht vorbei“
  - Einige einschlägige Gesetze und Regularien
    - National: HGB, BDSG, GdPdU, KontraG, GoB, IDW, KWG, ...
    - International: EU-Richtlinien, SOX, ...
  - Beispiele für gesetzliche und regulatorische Anforderungen:
    - Aufzeichnungs- und Aufbewahrungspflichten
    - Datenschutzbestimmungen
    - Einschränkung bei Einsatz bestimmter Technologien (z.B. Verschlüsselung)
    - Spezielle Anforderungen zuständiger Aufsichtsbehörden
    - Spezielle Vorgaben bei Outsourcing (z.B. Banken KWG 25a)
  - Allgemein:
    - Im Zweifel unbedingt juristische Expertise hinzuziehen

Regel 3: „Definieren Sie fachliche Zuständigkeiten, Rollen und Prozesse“

- Die Software soll letztlich Ihre Geschäftsprozesse unterstützen
  - Für Ihre eigenen Prozesse müssen Sie aber selber sorgen, die Software kann und soll Sie dabei aber unterstützen
  - Sie kennen Ihre Zuständigkeiten, Rollen und Prozesse am besten
  - Die IT kann daraus Nutzer- und Berechtigungs- und Rollenkonzepte ableiten:
    - „Need to know“-Prinzip
    - Unterschiedliche Rollen
    - Unterschiedliche Rechte, z.B. keine, lesend, schreibend, …
  - Beispiele für ggf. getrennte fachliche Rollen:
    - Auftragserfassung, -genehmigung, -freigabe, Inkasso, Kontrolle

Regel 4: Berücksichtigen Sie auch Anforderungen an die Nachvollziehbarkeit

- Nachvollziehbarkeit ist auch ein Sicherheitsziel
  - Fragestellung: Inwieweit müssen Daten, Zugriffe, Ergebnisse und Einzelschritte aus der Vergangenheit nachvollziehbar sein
  - Beispiele für entsprechende Fragestellungen:
    - „Wer hatte vor 7 Monaten mit welchen Rechten Zugriff auf die Auftragsdaten und wer hat was genau am Tag xy gemacht“
    - Rechtliche Anforderungen an die Nachvollziehbarkeit bestimmter Sachverhalte und Informationen, z.B. bilanzrelevante Daten (vgl. auch Regel 2)
    - Nachvollziehbarkeit von Zugriffen auf personenbezogene Daten
    - Aufbewahrungsfristen
  - Die IT kann daraus ein Logging- und Archivierungskonzept ableiten

Regel 5: Verfügbarkeit und Business Continuity können spezielle Sicherheitsanforderungen sein

- Zwei grundsätzliche Arten von Sicherheitsgefährdungen
  - Temporäre „Nicht-Verfügbarkeit“
    - „Wie lange kann ein Unternehmen temporär ohne gewisse Anwendungen oder Informationen seinen entsprechenden Geschäftsbetrieb aufrecht erhalten
  - Schlußfolgerungen für IT z.B.:
    - Wiederanlaufzeiten, Availabilitykonzept, USV, Wartungs- und Supportverträge
  - Totalverlust, Katastrophenfall
    - Totalverlust bestimmter Daten kann Existenz des Unternehmens gefährden
  - Schlußfolgerungen für IT z.B.:
    - Backup-Konzept, Datenlagerung, ggf. gar Notfalllokation

Regel 6: Die Sicherheitsanforderungen sollten bekannt, vereinbart und fixiert sein

- Gerade bei Sicherheitsanforderungen gibt es oft Mißverständnisse
  Im Prinzip zählen im Endeffekt nur klare Vereinbarungen
  - Häufig anzutreffende Ausgangssituation:
    - Die Umsetzung gewisser Sicherheitsanforderungen bzw. –eigenschaften wird oft „stillschweigend“ angenommen bzw. vorausgesetzt
    - Woher soll aber IT-Spezialisten per se die fachliche Bedeutung gewisser Daten klar sein ?
  - Schlußfolgerungen:
    - Auch Sicherheitsanforderungen müssen explizit benannt und definiert werden
    - Klärung je früher desto besser
    - Fixierung in Fachkonzept, Pflichten- oder Lastenheft
    - Klärungsprozeß führt auch zu realistischen Sicherheitsanforderungen

Regel 7: Erfüllung der Sicherheitsanforderungen sollte abnahmerelevant sein

- Schriftlich formulierte Abnahmekriterien auch für Sicherheit
  - Definition für nicht sicherheitstechnisch versierten Auftraggeber oft schwierig, mögliche konkrete Abnahmekriterien aber z.B.:
    - Eigene Prüfungen und Tests (soweit möglich, z.B. Rollenkonzept)
    - Vorlage und Abnahme von Testfällen, Prüfung der Erfüllung
    - Einbeziehung externer Kriterien, z.B. „Top Ten“ von OWASP
  - Ggf. unabhängige Prüfung bei sicherheitskritischen Anwendungen, Grundsätze und Empfehlungen dazu:
    - Nicht nur „Hacking“ sondern „White Box“-Vorgehen incl. Review
    - Angekündigt und im Einklang mit dem ursprünglichen Auftragnehmer

Regel 8: Beachten Sie Restrisiken und treffen Sie ggf. Vorkehrungen

- 100%ige Sicherheit nicht erreichbar und als Zielvorstellung auch nicht wirtschaftlich
  - Betrachtung und Analyse der wesentlichen Restrisiken
    - Es existieren auch bewusst in Kauf genommene Restrisiken
    - Präferiert: Aufstellung von vorneherein vereinbart
  - Alternative Überlegungen für Vorkehrungen bei Eintritt
    - Ggf. alternative nicht IT-technische Maßnahmen (z.B. Papier-Backup)
    - Technische Maßnahmen im späteren laufenden Betrieb (z.B. Security Patches)
    - Organisatorische Maßnahmen bei Eintritt eines der Risiken

Regel 9: Bedenken Sie auch Ihre eigenen Prozesse, organisatorischen Maßnahmen und Leute

- Nicht alle Sicherheitsmaßnahmen sind Sache der IT
  - Beispiel:
    - Sicherheitstechnisch ausgeklügeltes Berechtigungskonzept implementiert
    - Vergaben im späteren Betrieb aber quasi „auf Zuruf“ und ohne Überprüfung
  - Erforderliche eigene Maßnahmen:
    - Geregeltes (aber auch „lebbares“) Antrags- und Genehmigungsverfahren
    - Regelungen für Zuständigkeitswechsel, Austritt, Löschung
  - Allgemein:
    - Sicherheit läßt sich nur aufrecht erhalten, wenn sie tatsächlich „gelebt“ wird (Voraussetzung: auch „gelebt“ werden kann)
    - Usability versus Sicherheit
    - Schulungsmaßnahmen, Sensibilisierung, „Awareness“

Regel 10: Alles kann sich ändern, seien Sie darauf vorbereitet

- Veränderungen während oder vor allem auch nach Projektablauf
  - Treffen Sie Regelungen für erst später offensichtlich gewordene Mängel
    - Sicherheits-Updates bei Anwendung, aber auch bei Standardkomponenten
    - Nachbesserungspflichten, Wartungs- und Supportverträge
    - „Service Level Agreements“, insbesondere auch bei externem Betrieb
  - Halten Sie sich auf dem Laufenden, verschiedene Einflußgrößen können zu Neubewertungen (ggf. Folgebeauftragungen) führen, z.B.:
    - Neue fachliche Gegebenheiten, Neubewertung der Sicherheitsanforderungen
    - Neue oder veränderte gesetzliche Vorgaben