technische universität
dortmund

Vorlesung

Sicherheit:
Fragen und Lösungsansätze

Dr. Thomas P. Ruhroth

TU Dortmund, Fakultät Informatik, Lehrstuhl XIV

1) Sicherheit - Einleitung

Agenda

Inhalt

- Grundlegende Begriffe
- Einleitende Beispiele

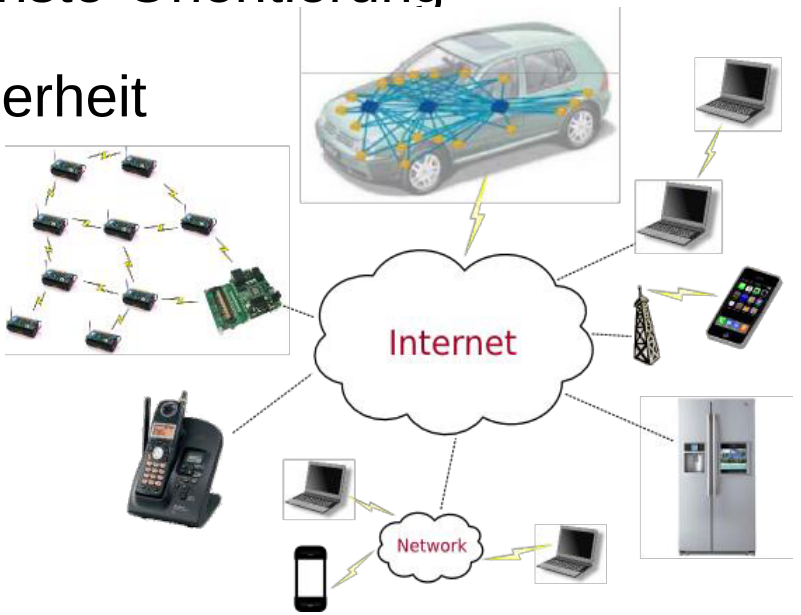
Lernziele

- Die Wichtigkeit von Sicherheit erkennen
- Hauptbereiche verstehen und erkennen
- Beispiele kennen und unbekannte Fälle klassifizieren können

IT-Trends

Mobilität, Vernetzung, Miniaturisierung, Dienste-Orientierung

- neue Herausforderungen für die IT-Sicherheit



Internet der nächsten Generation:

- Hochgradig verteilt und vernetzt
- Heterogen, dynamisch und kooperativ
- Internet der Dienste und Dinge

Ubiquitäre IT:

- Durchdringung unseres Alltags mit IT

Konsequenz: Anforderungen an die IT-Sicherheit steigen

Safety vs. Security

Safety versus Security (IT Sicherheit)

Safety:

- Erkennen und Abwehr von Störungen, die die korrekte Funktionalität, die Betriebssicherheit beeinträchtigen.
- Spezifikation der gewünschten Funktionalität und Erkennen von Abweichungen vom gewünschten Verhalten.
- Störungen treten in Software durch Programmierfehler auf.

Beispiele

- 1982: Absturz eines Lockheed F-117 Tarnkappenbombers: Die Steuerung des Höhenruders wurde mit der des Seitenruders vertauscht.
- Die NASA verlor 1962 ihre Venus-Sonde Mariner 1, und damit 80 Millionen US-Dollar, aufgrund eines fehlenden Bindestrichs im

Safety versus Security

Security (IT Sicherheit):

- Erkennen und Abwehr von insbesondere auch gezielten Angriffen:
- nach ISO (International Standards Organization)/IEC 2382-1:

  Minimierung der Verwundbarkeit von Werten und Ressourcen
  Bewahren eines Systems vor Beeinträchtigung und Missbrauch

Beispiele für Angriffe auf die IT-Sicherheit?

Wechselwirkungen: Security & Safety

- Sicherheitsverletzungen können Safety gefährden: Beispiel?
- Sicherheitsmaßnahmen können Safety gefährden: Beispiel?
- Safety-Verletzung kann Sicherheit gefährden: Beispiel?

Challenges für IT-Sicherheit

- Hochgradig verteilte IT-Systeme (SOA, Cloud, …)
  - Dezentrale Nutzung, Verarbeitung: Kontrollmöglichkeiten?
  - Viele Komponenten: Skalierbarkeit der Sicherheitsverfahren?
- Vielzahl von eingebetteten Komponenten, Sensoren
  - M2M-Kommunikation: Komponenten-Identifikation?
  - Ressourcenschwache Komponenten: Verschlüsseln?
- Stark vernetzte Systeme
  - Always-on: kaskadierende Schadensausbreitung?
  - Vielzahl von Angriffspunkten: Angriffstoleranz? Frühwarnung?
- Dynamische, Service-orientierte Systeme
  - Vertrauenswürdigkeit? Robustheit? Selbst-Organisation?

Begriffe und Bereiche

Grundlagen

Aufgabe: Schützenswerte Güter (Assets) identifizieren

- Gut wird als Objekt bezeichnet

Aufgabe: Einheiten, die auf Objekte zugreifen können, identifizieren

- derartige aktive Einheiten werden als Subjekte bezeichnet

Aufgabe: Zugriffe auf Objekte/Informationen beschränken

- Rechte zur Nutzung von Objekten, Rechte an Information festlegen

Aufgabe: nur autorisierte Zugriffe zulassen und ermöglichen

- Regelwerk (Policy) für Berechtigungen/Verbote festlegen

Aufgabe: Beschränkungen kontrollieren (Enforcement)

- Schutzmechanismen, Sicherheitsdienste und -protokolle festlegen

Schutzziele

Vorrausetzung: funktionale Korrektheit

Schutzziele sind das Gewährleisten der

(1) (Daten-) Integrität (engl. integrity)

(2) (Informations-) Vertraulichkeit (engl. confidentiality)

(3) Verfügbarkeit (engl. availability)

(4) Verbindlichkeit, Zurechenbarkeit (engl. accountability)

(5) Authentizität von Subjekten, Objekten (engl. authenticity)

(6) Privatheit, Anonymität, Nicht-Verfolgbarkeit (engl. privacy)

idR. wird eine Kombination mehrerer Schutzziele gefordert

Einordnung des Begriffs Trust, Vertrauenswürdigkeit?

Datenintegrität

Definition Datenintegrität: Schutz vor unautorisierter und unbemerkter Modifikation von Daten

Maßnahmen zur Gewährleistung: u.a.

- Regeln für zulässige/unzulässige Datenänderungen
  - Wer (Subjekt) darf was (Rechte) unter welchen Bedingungen mit welchem Objekt (Asset) tun
- Vergabe von Zugriffsrechten und Kontrolle: z.B. r, w, x
- Isolierung: Schutzdomänen, Sandboxes, VMs,
- Manipulationserkennung: Prüfwerte, digitales Watermarking

Konkrete Mechanismen aus der Praxis?

Informationsvertraulichkeit

Definition Informationsvertraulichkeit: Schutz vor unautorisierter Informationsgewinnung

Maßnahmen:

- Regeln für zulässige/unzulässige Informationsflüsse
  - Wer darf auf welche Informationen zugreifen, bzw. davon Kenntnis erlangen
- Verschlüsselung von Daten
- Informationsflußkontrolle: Klassifizieren von Objekten und Subjekten
- speziell: Confinement-Problem: Verdeckte Kanäle, Seitenkanäle

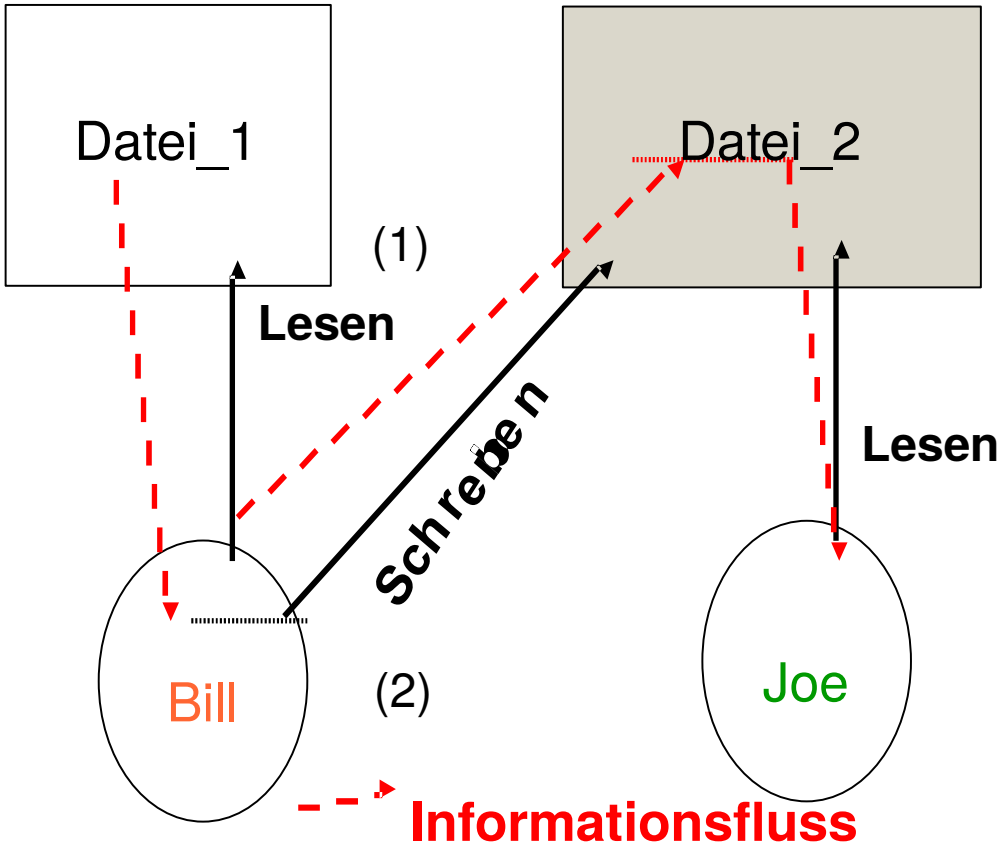
Beispiel?

Konkrete Mechanismen aus der Praxis?

Beispiel: Informationsflußkontrolle

Beispiel: Informationsflußkontrolle häufig wünschenswert

- Forderung: Subjekt Joe darf keine Kenntnis über vertrauliche Informationen erlangen, Bill aber sehr wohl!



Annahme:
Subjekt Chef schreibt vertrauliche Informationen in Datei_1 (Objekt),
Regeln: Joe hat kein Recht, auf Datei_1 zuzugreifen!

Annahme:
Bill liest die Information aus Datei_1 und schreibt sie in Datei_2
⇒ **Forderung wird verletzt**

Verfügbarkeit

Definition Verfügbarkeit: Schutz vor unbefugter Beeinträchtigung der Nutzbarkeit und der Funktionalität/Dienste etc.

Maßnahmen: Regeln zur Aufzeichnung und Überwachung

- welche Zugriffe auf welche Objekte, wann mit welchem Ressourcenverbrauch (z.B. Speicher) aufzeichnen
- Festlegen von Schwellwerten, z.B. Überlast

Bem.: Regelungen sind Verpflichtungen (engl. obligation) etwas zu tun (im Gegensatz zu den Berechtigungen/Verboten)

Konkrete Mechanismen aus der Praxis?

Verbindlichkeit, Zurechenbarkeit

Definition Verbindlichkeit, Zurechenbarkeit: Schutz vor unzulässigem Abstreiten durchgeführter Handlungen

Maßnahmen: festlegen, was verbindliche Aktionen sind

- Nicht-Abstreitbarkeit: nicht immer Rechtsverbindlichkeit erforderlich
- Kopplung von Aktionen mit Subjekt, das Aktion ausführt: Signatur
- Protokollieren von Aktionen und Zeitpunkten, Log-Dateien auf Betriebssystem-Ebene protokollieren sicherheitsrelevante Aktionen
- Beweissicherungen durchführen (u.a. für forensische Analysen)

Konkrete Mechanismen aus der Praxis?

Authentizität

Definition Authentizität: Nachweis der Echtheit und Glaubwürdigkeit der Identität eines Objekts/Subjekts

Maßnahmen:

- Regeln zu Vergabe von eindeutigen Identifikationen von Subjekten, Objekten: Passworte, Schlüssel, Biometrie, Smartcards, …
- zunehmend nicht nur Identifikatoren sondern auch Attribute, die das Subjekt charakterisieren
- Verfahren zum Nachweis der Korrektheit der Identität
  - Ausstellen und Prüfen von Zertifikaten, Credentials, Token
  - Challenge/Response Protokolle: Frage/Antwort Abläufe

Konkrete Mechanismen aus der Praxis?

Privatheit

Definition Privatheit: Schutz der personenbezogenen Daten, Schutz der Privatsphäre, Gewährleistung des informationellen Selbstbestimmungsrechts

Maßnahmen: Regeln zu Datenvermeidung und zur Datensparsamkeit

- Festlegung der Zweckbindung der erhobenen Daten
- Datenaggregation: k-Anonymity Verfahren
- Pseudonyme: Identität ist einer Trusted Third Party bekannt (‚Notar‘)
- Nicht Verfolgbarkeit, Non-Tracebility: wechselnde Pseudonyme

Sicherheitsregeln Security-Policies

Sicherheitsregeln, Security-Policies

- Festlegen der Schutzziele und der Menge von technischen und organisatorischen Regeln und Verhaltensrichtlinien
- Festlegen von Maßnahmen zur Gewährleistung der Schutzziele, um ein angestrebtes Sicherheitsniveau zu erzielen
- Festlegen von Verantwortlichkeiten und Rollen

Beispiel: Passwort-Policy (Auszug)

- All system-level passwords (e.g., root, enable, NT admin) must be changed on at least a quarterly basis.
- All user-level passwords (e.g., email, web, desktop computer, etc.) must be changed at least every six months.
- Passwords must not be inserted into email messages or other forms of electronic communication.

Schwachstelle, Bedrohungen, Risiken, Angriffe

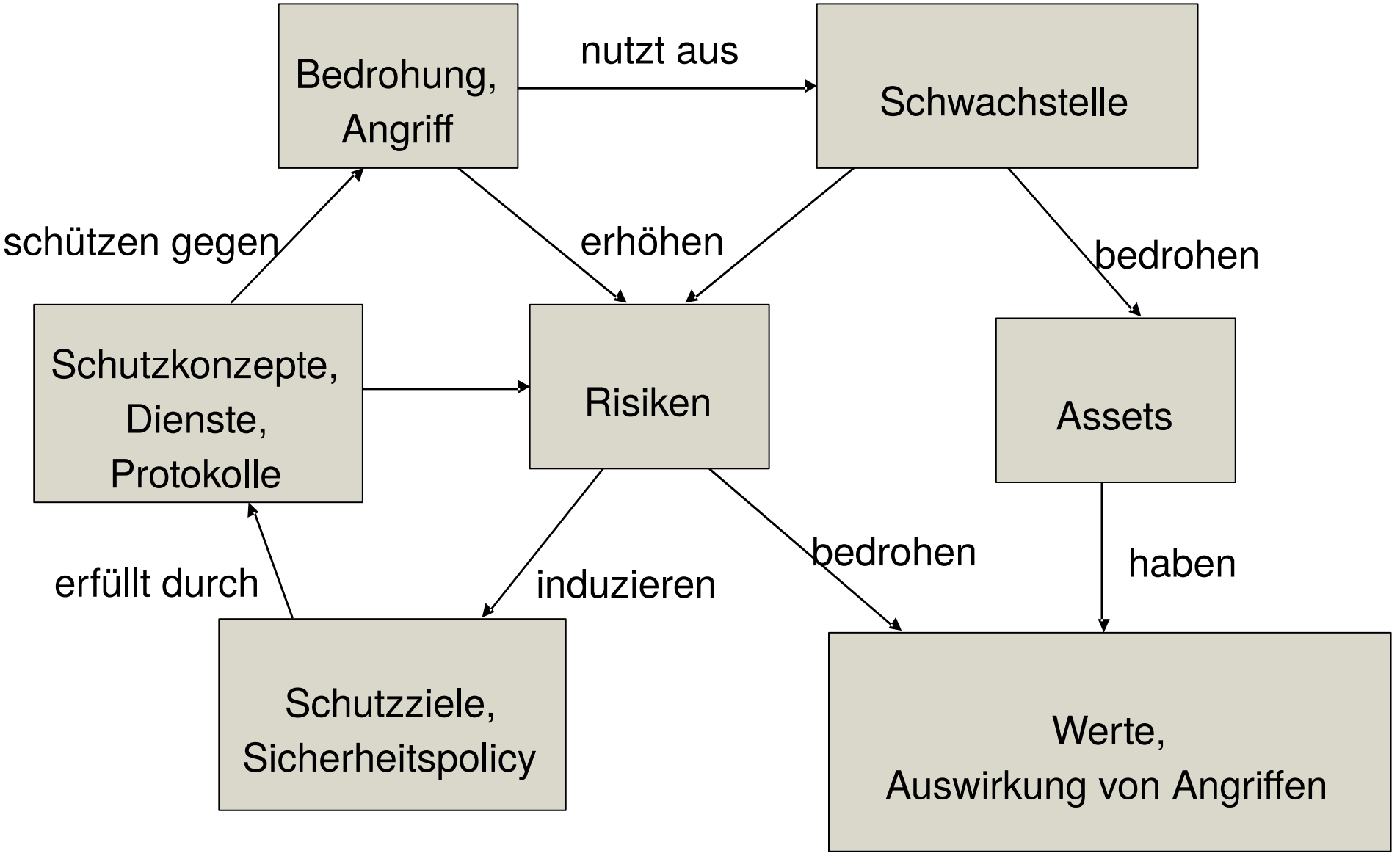
Eine Schwachstelle (Vulnerability) (u.a.) im Code, im Protokoll, ermöglicht es, dass die Sicherheitsdienste des Systems umgangen, geändert oder getäuscht werden können.

- Bedrohungen (Threat) ergeben sich aus möglichen Angriffen, die eine oder mehrere Schwachstellen eines Systems ausnutzen, um ein oder mehrere Schutzziele zu gefährden.
- Das Risiko *R* (Risk) einer Bedrohung ist die Wahrscheinlichkeit *E* des Eintritts eines Schadensereignisses und die Höhe des potentiellen Schadens *S*, der daraus resultieren kann:
  $R = E \cdot S$

Angriffsziele

- Unter einem Angriff (Attack) verstehen wir einen nicht autorisierten Zugriff auf ein Asset
  - Ein Angriff nutzt dazu eine Schwachstelle des Systems aus
  - weit verbreitete Schwachstellen: u.a.
    - mangelhafte Identitätsprüfung: Spoofing-Angriffe
    - fehlende Eingabe-Validierung: Buffer-Overflows, XSS, ...
- **Angriffsziele** und Angriffsklassen
  - Netze: Sniffen, Spoofen, DoS, …
  - Web-Anwendungen und Datenbanken**:** XSS, SQL-Injection
  - Server, PC: Buffer-Overflow, Viren, Würmer, Trojaner
  - Benutzer: Phishing, Spamming, Social Engineering

Zusammenhänge zwischen den eingeführten Begriffen



Sicherheitsprobleme
Beispiele

Ausgewählte Sicherheitsprobleme

Angriffsziele:

- Gefährdung von einem oder mehreren Schutzzielen

Häufige **Ansatzpunkte:**

1. mangelhafte Identitätsprüfung: Spoofing-Angriffe
   Zustandsinformationen manipulieren: z.B. Cache-Poisoning
2. nicht korrekt überprüfte Eingaben: Buffer-Overflows, Cross-Site-Scripting, SQL-Injection, ...
3. universelle Interpretierbarkeit: Daten, Code: Einschleusen von Malware (Viren, Würmer, Trojaner)
4. Social Engineering: Nutzer ist häufig schwächstes Glied in der Sicherheitskette: Viren, Würmer, Trojaner-Angriffe

Mangelhafte Identitätsprüfungen

Angriffe auf allen Schichten des OSI-Modells anzutreffen

- Schicht 2:
  - ARP-Spoofing
- Schicht 3 und höher:
  - IP-Address-Spoofing
- Schicht 4, 5 (und höher):
  - Session-Hijacking …
- Schicht 7: zugeschnittene Spoofing-Angriffe: u.a.
  - DNS-Spoofing
  - gespoofte E-Mail-Absenderadressen
  - Web-Spoofing

Nicht korrekt überprüfte Eingaben

Beispiel: Buffer-Overflow-Exploits

- **Ziel:** einschleusen von Code (Viren, Würmer, Trojaner), Verändern von Daten (z.B. ‚Umbiegen' von Funktions-Pointern)
- BO: „Schwachstelle des Jahrzehnts" (Bill Gates, in den 90ern)
- **Ansatz:** Ausnutzen von Programmierfehlern!

Vorgehen: (ganz allgemein)

- Überschreiben des Speicherbereichs, die für die Werte einer Variable (z.B. String, Array, Integer) vorgesehen ist, mit zu großen Werten, so dass der reservierte Bereich (das ist der ‚Buffer') überläuft (overflow).

Ursache für erfolgreichen Überlauf:

- Ungeprüfte Übernahme von Eingaben/Werten
- häufig bei Programmen/Diensten, in denen Daten über Eingaben in eine Variable eingelesen werden, ohne dass die Größe des Eingabewerts überprüft wird

‚Generationen' von Buffer-Overflow-Angriffen:

- **erste Generation**: Ausnutzen von fehlerhaft programmierten Operationen zum Kopieren von Zeichenketten, z.B. strcpy
- **zweite Generation**: Überlauf einzelner Integer-Zahlen, oder Nutzung von Schleifen, die nicht korrekt terminieren z.B. das zeichenweise Kopieren einer URL.

Betroffene Bereiche

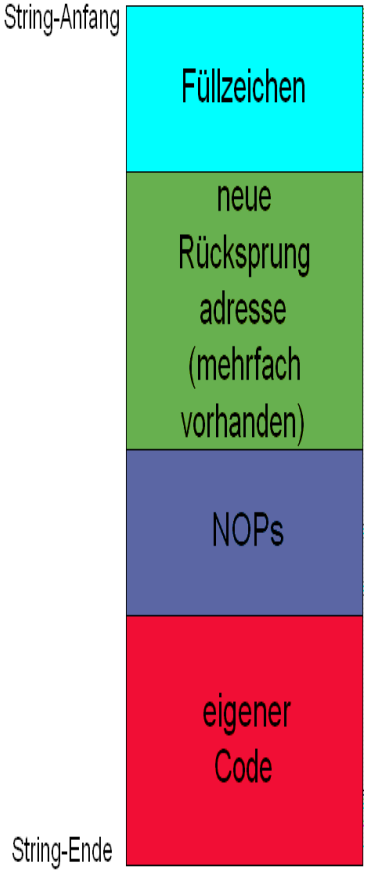
Stack-Bereich des Prozessadressraums: häufigster Angriffsbereich, Stack-Smashing-Angriffe

- Heap-Bereich des Adressraums: Angriffe u.a.:
  - Überschreiben von (Funktions-)Zeigern,
  - Überschreiben von mit *malloc()* initialisierten Speicherbereichen mit eigenem Code, ohne Speicherschutzverletzung zu produzieren
  - Effekte: z.B. globale Variable wie

    char[] tmp = "/tmp/prog_swap" mit

    char[] tmp = "/root/.ssh/authorized_keys"

    überschreiben und mit beliebigen Inhalten füllen
- Register-Bereiche, Flags (z.B. im Kernel, Zugriffs-Modi)

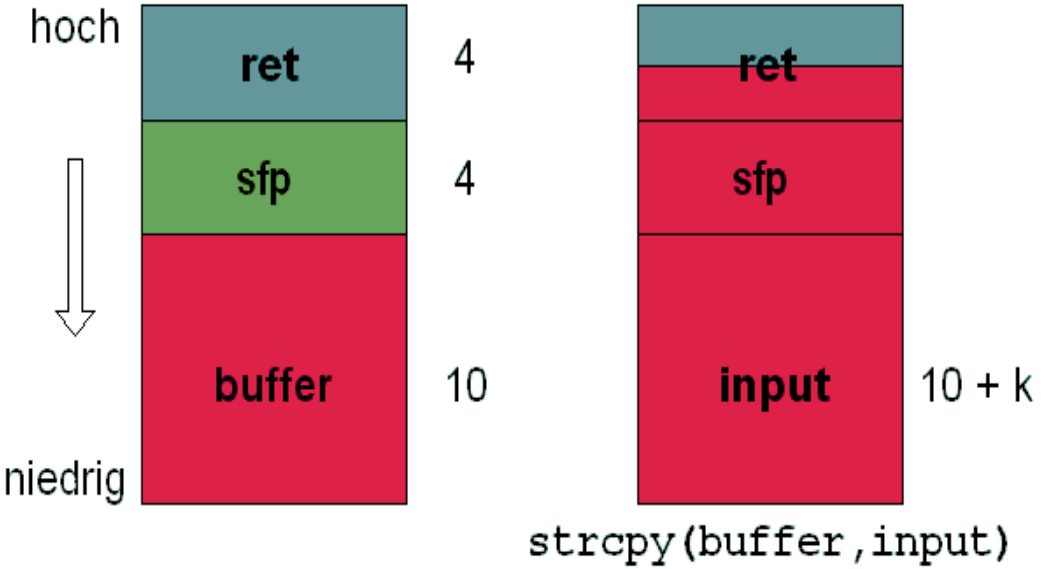


Beispiel: Buffer-Overflow-Angriff

Beispiel: Buffer-Overflow-Angriff:

Beispiel-Code:

```
char input[] = "Windsurfing Hawai
 //sizeof(input) = 18
char buffer[10];
strcpy(buffer,input);
```



Effekt:

- *strcpy()* kopiert Input in Variable *buffer,* ohne Prüfung der Bereichsgröße
- Überschreiben der Rücksprungadresse,
- falscher Wert: beim Return in Instruction-Register geladen

Gefährdungen durch erfolgreiche BO-Angriffe

Konsequenz einer veränderten Rücksprungadresse:

1. überschriebene Rücksprungadresse enthält keine sinnvolle Adresse: Segmentation Fault, ggf. bis hin zum Systemabsturz (Verfügbarkeit!)
2. Rücksprungadresse enthält sinnvolle Adresse: Programm macht dann ‚irgendwas', d.h. es verhält sich nicht mehr gemäß Spezifikation (Integrität, Vertraulichkeit, Authentizität!!)
3. Rücksprungadresse enthält sinnvolle Adresse von auszuführendem Maschinencode, der vom Angreifer auf den Stack platziert wurde: Trojaner, Virus ...

Gegenmaßnahmen

Techniken zur sicheren Programmierung nutzen:

- Eingabe-Filterung:
  - Bereichsgrenzen prüfen,
  - Spezifikation mittels regulärer Ausdrücke
- Typsichere Sprachen wie Java verwenden: durch Laufzeitsystem und Compiler erfolgen Bereichs- und Typprüfungen

- Für C, C++: Verwendung spezieller Bibliotheken:
  - z.B. Libsafe (Linux): Wrappen von Standard- C-Bibliotheksaufrufen durch Libsafe-Aufrufe

http://www.research.avayalabs.com/gcm/usa/en-us/initiatives/all/nsr.htm

Falls **Neu-Übersetzung des Quellcodes** möglich:

- z.B. Programm StackShield für Linux-Systeme: sichert bei jedem Funktionsaufruf die Returnadresse und korrigiert sie bei Bedarf
- Stack-Smashing-Protector: GNU C-Compiler seit Version 4.1 fügt Kontrollzeichen (canary) direkt hinter Rücksprung-adresse ein, prüft Canary vor Rücksprung, schreibt Warn-meldung in Syslog und terminiert Programm, falls Änderung erkannt wurde

  (Stack-Cookie seit Windows Vista/Server 2003 analog)

Bewertung des Ansatzes?

Bemerkung: Stack-Überwachungsprogramme liefern natürlich auch keinen Schutz vor Heap-Overflow-Exploits

Betriebssystem-unterstützter Schutz:

- Z.B. unter Sun-Solaris konfigurierbar, dass im Stack-Segment kein Code ausführbar ist (Stack als non-executable)
- Ähnliche Patches für Linux http://www.openwall.org,
- Und auch für Windows XP/2003

  http://www.lumension.com/Press---Events/Press-Releases/SecureWave-Releases-SecureStack-v3-0.aspx

Aber: **dieser** Lösungsansatz hilft nichts, wenn durch eingeschleusten Code bereits installierter Code von DLLs oder von Bibliotheksdiensten ausgeführt wird!

- weitere Ansätze: Compartment-Mode Systeme (z.B. Trusted OS): Isolierung von Speicherbereichen, Eindämmen der Schadensausbreitung

Lessons Learned

Buffer-Overflow ist ein fast allgegenwärtiges Problem,

- tritt zwar idR. bei der Verarbeitung von Prozeduraufrufen auf dem Stackbereich des Prozess-Adressraums auf,
- kann aber auch andere Datenbereiche betreffen!
- Erfolgreiche BO-Angriffe können alle Schutzziele gefährden!
- BOs ergeben sich aus unsicherer Programmierung!

Benötigt:

- Regeln zur sicheren Programmierung einhalten, typsichere Sprachen verwenden, Sicherheitsanalyse des Codes
- in Entwicklung: neue, erweiterte Analysetools, die bereits zur Compilezeit mögliche BO-Schwachstellen erkennen

Beispiel: SQL-Code-Injektion

Ziel: Einschleusen von fremden SQL-Code in die Datenbank

Angriffsfootprint:

- Angreifer konstruiert manipulierten Eingabestring
- Eingaben enthalten SQL-Kommandos
- Zugriff auf SQL-Datenbank mit Rechten des Web-Servers

Problem: Eingabedaten eines Benutzers werden beim Web- Server als Kommando oder Anfrage (z.B. Datenbank-Query) verstanden und entsprechend ausgeführt

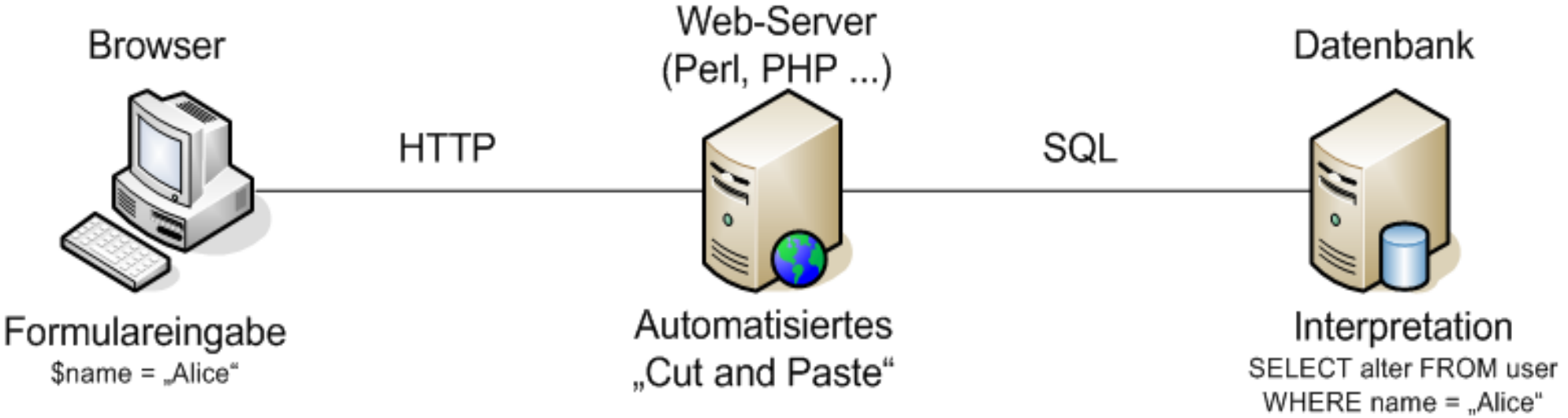
Impact: Angreifer erhält z.B. Zugriff auf Daten in Datenbanken, die vom Server betrieben werden, Daten können gelesen, manipuliert und es können komplexe Datenbank-Befehle initiiert werden.

Typischer Webservice

Benutzer gibt Daten in einem Webformular ein

- Server setzt diese Daten in SQL-Statements ein

- Schwachstelle:
  - Server setzt SQL-Anfrage aus Textblöcken zusammen, sendet diese ungeprüft als String weiter zum DB-Server



Beispiel: SQL-Code-Injection Attacke

Nutzer gibt spezifische Daten in einem Webformular ein

- Herkömmliche Abfrage:

  $name = "**max**"

  SELECT age FROM user WHERE name=`**max**`

- Angriff (Einschleusen von Code):

  $name = "**max**`; SELECT salary FROM personal WHERE name=`fritz"

- Zusammengesetzte Anfrage wird interpretiert zu:

  SELECT age FROM user WHERE name = `**max**`;

  SELECT salary FROM personal WHERE name=`fritz`

Malware-Trends

Organisierte Kriminalität im Internet:

- Kommerzielles Erstellen, Verbreiten und Einsetzen von Malware
- Professionelle Hacker
- Schattenwirtschaft mit Gewinnen in Millardenhöhe

Malware

Modularer Aufbau mit vielfältigen Einsatzmöglichkeiten

- Backdoor, Keylogger, Spionage (Dateiübertragung usw.)
- Integration in Bot-Netz für Spam-Versand, DDoS-Angriffe, Hosten von illegalem Content
- Automatische Updates (oft mehrmals täglich), Nachrüsten von Schadroutinen und Tarnmechanismen
- **Unterteilung in Wurm/Trojaner/Spyware/Bot kaum mehr möglich!**

Social Engineering

- Trend: maßgeschneiderter Code für kleinen Kreis von Opfern
- Gezielte Angriffe gegen Behörden, Unternehmen (Industriespionage) und Personen (Identitätsdiebstahl)
- Verschlüsselung des eigenen Codes,
- Erkennen von Sandbox-Umgebungen zur Analyse, Ändern des eigenen Verhaltens in solchen Fällen

Social Engineering

- Präparierte Webseiten, E-Mails, Instant Messenges oder Datenträger, die in sozialen Kontext des Opfers passen
- Z. B. „Verlieren“ von USB-Sticks mit schadhaftem Content
- Opfer klickt auf Link oder öffnet „harmloses“ PDF, wobei wieder Lücken in den Anwendungen zur Infektion führen

Quelle: Lagebericht der IT-Sicherheit 2011, BSI.
https://www.bsi.bund.de/DE/Publikationen/Lageberichte/lageberichte_node.html

Sicherheitnormungen und Standards

Norm/Standards: technisch

- technische Übereinkünfte oder Verordnungen (Normen), die sich in der Praxis eine breite Akzeptanz verschafft haben,
- De-facto- oder Quasi-Standards: vereinheitlichte und abgestimmte Regelwerke und Spezifikationen, die sich z.B. in der Praxis entwickelt und etabliert haben
- Norm: allseits rechtlich anerkannte und durch ein Normungsverfahren beschlossene, allgemeingültige sowie veröffentlichte Regel zur Lösung eines Sachverhaltes.

Ziel:

vereinheitlichte Spezifikation, um Interoperabilität, Kompatibilität, und Vergleichbarkeit von Produkten zu erreichen

Einige wichtige Organisationen:

- **ISO:** International Standard Organisation
- **NIST:** National Institute of Standards and Technology: USA
- **W3C:** World Wide Web Consortium: Web Standards: HTML, XML
- **OASIS:** Organization for the Advancement of Structured Information Standards: u.a. XACML, UDDI, WS-Security
- **BSI:** Bundesamt für Sicherheit in der Informationstechnik: entwickelt Grundschutzstandards und technische Richtlinien

Sicherheitsstandards: u.a.

- Informationssicherheitsmanagement (ISM)
- Sicherheitsevaluation
- Spezielle Sicherheitsfunktionen

Infomationssicherheitsmanagement Quelle: BITKOM Leitfaden

ISO/IEC 27001	Information security management systems – Requirements Informationssicherheits-Managementsysteme - Anforderungen
ISO/IEC 27002	Code of practice for Information security management Leitfaden zum Informationssicherheitsmanagement
ISO/IEC 27006	Requirements for bodies providing audit and certification of information security management systems Anforderungen an Stellen, die Auditierung und Zertifizierung von Informationssicherheitsmanagementsystemen bereitstellen
IT-GS	IT-Grundschutz
ISO/IEC 18043	Selection, deployment and operation of intrusion detection systems (IDS) Auswahl, Einsatz und Betrieb von Systemen zur Erkennung des Eindringens in Netze und Systeme (IDS)
IOS/IEC 15816	Security information objects for access control Sicherheitsobjekte für Zugriffskontrolle
ISO/IEC 24762	Security techniques – Guidelines for information and communications technology disaster recovery services
ISO/IEC 25777	Information and communications technology continuity management. Code of practice
Risikomanagement	
MaRisk	Mindestanforderungen an das Risikomanagement für Banken
ISO/IEC 27005	Informaion security risk management Informationssicherheits-Risikomanagement

Sicherheitsevaluation: Common Criteria

ISO/IEC 15408 (CC)	Evaluation criteria for IT security (Common Criteria) Evaluationskriterien für IT-Sicherheit
ISO/IEC TR 15443	A framework for IT security assurance Rahmenrichtlinien für Sicherung von IT-Sicherheit
ISO/IEC 18045	Methology for IT security evluation Methodik zur Evalution von IT-Sicherheit
ISO/IEC TR 19791	Security assessment for operational systems Bewertung der Sicherheit von Systemen im Betrieb
ISO/IEC 19790 (FIPS 140-2)	Security Requirements for Cryptographic Modules Anforderungen an kryptographische Module
ISO/IEC 19792	Security evaluation of biometrics Evaluierung der IT-Sicherheit biometrischer Technologien
ISO/IEC 21827 (SSE-CMM)	System Security Engeneeting – Capability Maturity Model Modell der Ablaufstauglichkeit (auch ISO 21827)
ISO/IEC 24759	Test requirements for cryptographic modules Prüfungsanforderungen für kryptographische Module

Spezielle Sicherheitsfunktionen

Verschlüsselung	
ISO/IEC 7064	Check charakter systems - Prüfsummensysteme
ISO/IEC 18033	Encryption algorithms - Verschlüsselungsalgorithmen
ISO/IEC 10116	Modes of operation for an n-bit block cipher - Betribsarten für einen n-bit-Blockschlüssel-Algorithmus
ISO/IEC 19772	Data encapsulation mechanisms - Daten verkapselnde Mechanismen
Hash-Funktionen und andere Hilfsfunktionen	
ISO/IEC 10118	Hash function - Hash-Funktionen
ISO/IEC 18031	Random bit generation - Erzeugung von Zufallszahlen
ISO/IEC 18032	Prime number generation - Primzahlerzeugung
Authentifizierung	
ISO/IEC 9798	Entity authentication - Authentisierung von Instanzen
ISO/IEC 9797	Message Authentication Codes (MACs) - Nachrichten-Authentisierungscodes (MACs)
Schlüsselmanagement	
ISO/IEC 11770	Key management - Schlüsselmanagment

Nächste Woche:
Mathematische Grundlagen