Sichere Koprozessoren für Anwendungen im E-Commerce

Michael Kropfberger

13. Dezember 2000

Einführung

- Reales Leben:
  - kein Geld unterm Polster
  - kein blindes Vertrauen gegenüber Verkäufern
- E-Commerce:
  - halbherzige Ansätze in Software
  - zu viel Anwender-Interaktion

Überblick

- Gefahren und deren Beseitigung durch sichere Koprozessoren
- Familie der sicheren Koprozessoren
- Hardware- und Softwareanforderungen
- zwei Beispielimplementierungen
- High-Level Applikationen für sichere Koprozessoren

Gefahren

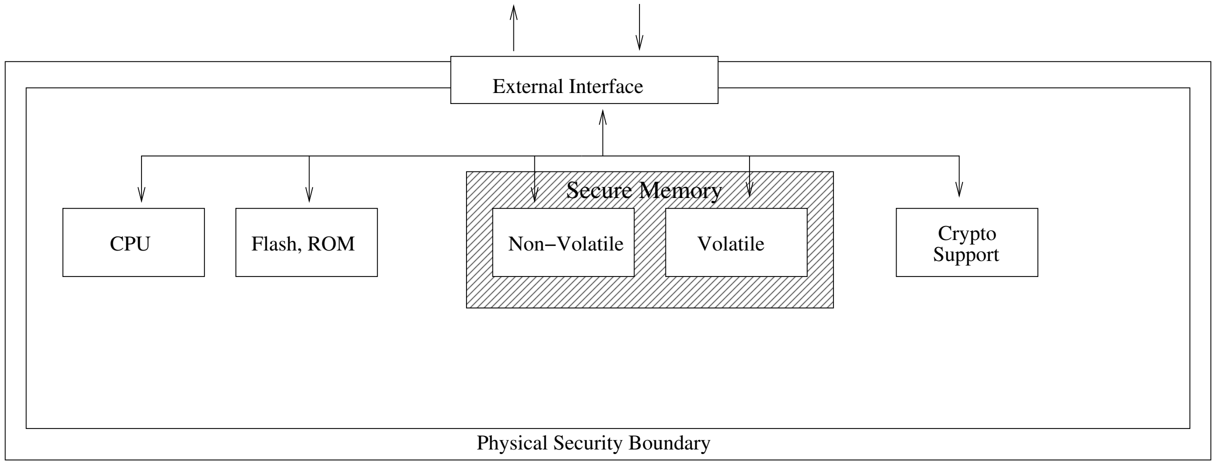
- physisch ungeschützte Rechner
  - Vandalen oder Hacker
  - Änderung von OS, Software und Dateien, “back doors”
  - Anzapfen der Geräte $\rightsquigarrow$ Passwörter oder Transaktionen
- Softwarefehler
  - Speicherschutz
  - Zugriffsverweigerung für OS auf Schlüssel

Gefahren[2]

- Unehrliche Angestellte, Vertragsbedienstete und Kunden
  - Angestellte: Zugriff auf Informationen und Hardware
  - Vertragsbedienstete: fühlen sich noch weniger an Firma gebunden
  - Kunden: Zugriff auf Intranet, Preise können auch für Konkurrenz interessant sein!

- Backup und Wiederherstellung
  - zu grobkörnige Zugriffsrechte: Angestellte machen Backups von lesbaren Daten
  - Backups müssen weitverteilt (auch ausserhalb der Firma) gelagert werden $\rightsquigarrow$ Diebstahl, Veränderung

Sichere Koprozessoren



- secure memory enthält geheime Schlüssel oder Daten
- Schutzhülle, bei Angriff wird "geschützter Speicher" gelöscht
- spezielles OS $\rightsquigarrow$ Crypto-Unterstützung

Gefahren[3]

- physisch ungeschützte Rechner
  - bei Angriff wird der “geschützte Speicher” gelöscht
- Softwarefehler
  - Speicherschutz, getrennte Adressräume
  - Zugriffsverweigerung für OS auf Schlüssel
  - Cryptographische Basisalgorithmen in Hardware

Gefahren[4]

- Unehrliche Angestellte, Vertragsbedienstete und Kunden
  - extra CPU und Programmierbarket
  - erweiterte Rechteverwaltung
  - Verschlüsselung
- Backup und Wiederherstellung
  - Verschlüsselung
  - Schlüssel wird weltweit repliziert
  - Schlüssel nur unter bestimmten Umständen verwendbar

Familie der sicheren Koprozessoren: Faktoren

Aufgliederung durch Faktoren wie:

- Sicherheit
- Rechenleistung
- Kryptographieleistung
- Kosten

Familie der sicheren Koprozessoren: Chip Card

- Chip Card (= Smart Card)
  - 8-bit CPU, 512 bytes RAM, 8K ROM, 8K EEPROM
  - Minimale Hardwareunterstützung für Kryptographie
  - ⊖ Programmierung: beim Hersteller, Brennen ins ROM
  - ⊖ Keine getrennten Speicherbereiche
  - ⊖ nur 9600 bit/sec Datenfluss zur Aussenwelt
  - ⊖ Code ist aus Platzmangel hand-optimierter Assembler
  - ⊕ Preis, physische Stabilität und Transportierbarkeit

Familie der sicheren Koprozessoren: Persönliche Token

- Persönliche Token
  - zB. PCMCIA Karten
  - mehr Speicher
  - mehr CPU Leistung
  - höhere Transferrate zur Aussenwelt
  - ⊖ meistens nicht programmierbar
  - ⊕ kann auf physische Attacken reagieren

Familie der sicheren Koprozessoren: Crypto-Beschleuniger

- Crypto-Beschleuniger
  - 2 MB EEPROM, 512K RAM
  - kleine Microprozessoren mit Crypto-Hardware Support
  - $\oplus$ “stream cyphering”, zB Dateisysteme, Speicherbereiche,...
  - geheime Schlüssel werden im “geschützten Speicher” gespeichert
  - $\ominus$ Programmierbarkeit steigt mit Absatzmenge :)
  - $\oplus$ besonders stabil (gegen Attacken) verpackt
  - $\ominus$ bei erfolgreicher Attacke $\leadsto$ alles zugreifbar

Familie der sicheren Koprozessoren: High-End Geräte

- High-End Geräte
  - Megabyteweise RAM und FLASH
  - schnelle CPU (eg. Intel 486)
  - Crypto-Chips
  - Echtzeituhr
  - Zufallszahlengenerator
  - ⊕ Programmierbarkeit, getrennte Adressräume
  - ⊖ Kosten, Komplexität

Benötigte Hardwareanforderungen

- Zwei Möglichkeiten zur Antwort auf Attacken:
  - **Aktiv** Software selbst, bei ständiger Stromzufuhr
  - **Passiv** physische oder chemische Standfestigkeit der “Black Box” (auch durch Sprengstoff)
- Faktoren für Anbindung and den Wirt:
  - Einfachheit der Installation
  - Platformunabhängigkeit
  - Geschwindigkeit des Interfaces
  - sicherheitsgeprüfte Gerätetreiber
  - $\leadsto$ PCMCIA, chip-card oder PCI-bus????

Benötigte Hardwareanforderungen[2]

- Kosten und Haltbarkeit
  - Standardteile: billig, aber niedriger Spezialisierungsgrad
- Exportierbarkeit
  - hohe Verschlüsselungssicherheit
  - zB U.S. Exportgesetz

Benötigte Softwareanforderungen

- Entwicklung
  - billige und einfache Prototypen?
  - Sicherheitsrelevante Daten/Code and Hersteller?
  - Codeoptimierer, Debugger, Testumgebungen?
- Installation
  - sichere Auslieferungskanäle? (Post, Wiederverkäufer...)
  - Identifizierung von “gutartigen” Updates beim Gerät?

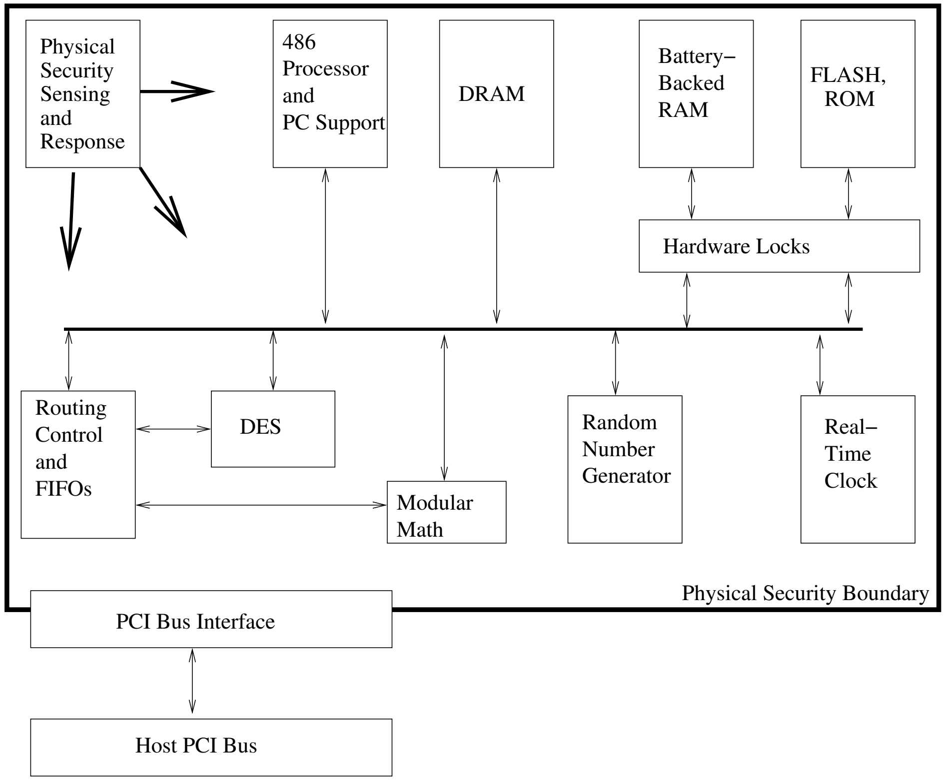
Benötigte Softwareanforderungen[2]

- Softwarewartung
  - "remote" multi-updates oder Einzelarbeit vor Ort von geschultem, geprüften Personal?
  - was passiert mit gespeicherten Daten wärend eines Updates?
  - Atomarität? $\rightsquigarrow$ Zugriff/Zerstörung von Geheimnissen bei unvollständigen Updates? $\rightsquigarrow$ Update der atomaritätsüberprüfenden Software selbst?

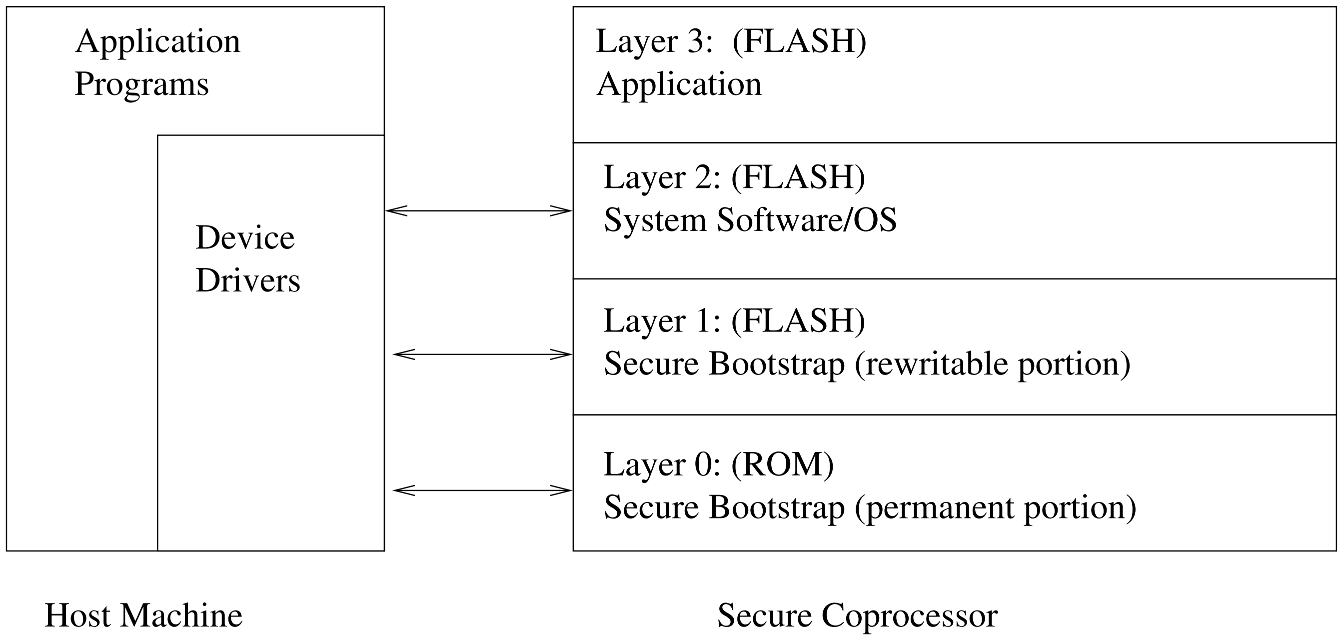
- Mehrere Köche...
  - OS sowie unterschiedlichste Applikationen nicht aus einer Hand
  - Speicherschutz auch zwischen Apps und OS!

IBM Forschungsprojekt

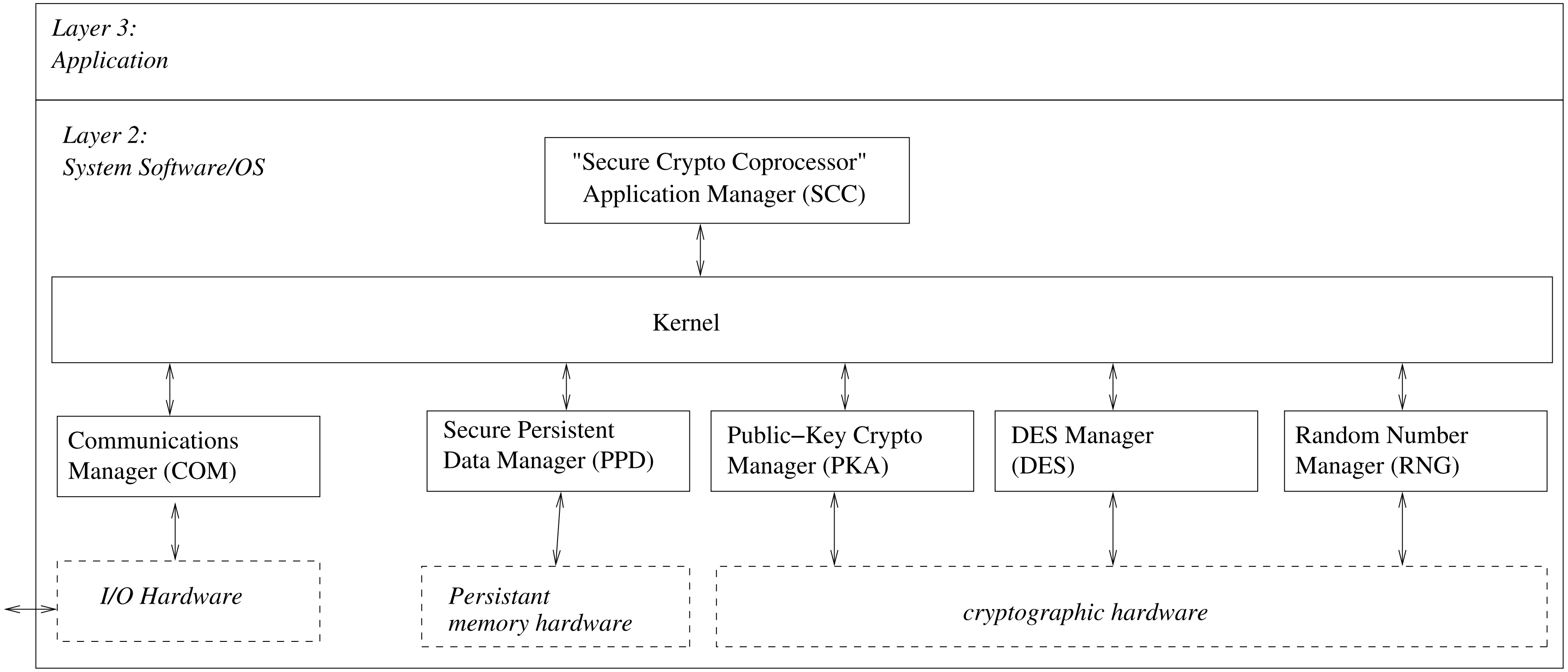
Intel 486, einige MB RAM, Crypto-Hardware, reagiert aktiv auf Attacken



IBM: 4-Schichtenarchitektur



IBM: Kernel-Schicht mit Managern



IBM: Details

- Kommunikation
  - verschlüsselte Kanäle zwischen Apps
  - Hardware FIFOs $\rightsquigarrow$ DMA Transfers im Hintergrund
- geschützter und persistenter Speicher
  - **Integrität** Daten werden durch Errors oder Attacken nicht verändert
  - **Geheimhaltung** vor Unbereichtigten, auch bei Attacken
  - **Atomarität** keine inkonsistenten Zustände

IBM: Details[2]: Sicherer Speicher

- FLASH
  - Grosser Speicher in Blockgrössen zwischen 4 und 64 KB
  - langsames Schreiben
  - bei Attacken nur zu langsam löschbar
  - und das nur mit Stromversorgung
- Batteriegestütztes RAM (BBRAM)
  - wenig Speicher, wahlfreier Zugriff
  - zum Löschen muss nur die Stromversorgung abgenommen werden
- $\leadsto$ BBRAM enthält Schlüssel, mit dem FLASH verschlüsselt ist
  bei Attacke wird BBRAM gelöscht

IBM: Details[3]

- Crypto Support
  - Hardware Support für Public Key und DES Verschlüsselung
  - Hardware FIFOs und DMA für “stream cypher”
- Hardware Zufallszahlengenerator
  - thermisches Rauschen wird in Strom von Zufallsbits umgewandelt
- Software Zufallszahlengenerator([7] und [8])
  - bessere Performanz
  - Wiederholbarkeit für Tests (gleicher “seed”)

M$ Forschungsprojekt “Dyad”

- grundsätzlich gleicher Aufbau wie bei IBM
- Checksummen für OS und Applikationen im gesicherten Speicher
- Crypto-Paging und -Versiegelung
  - Speicherseiten werden verschlüsselt $\rightarrow$ Hauptspeicher/Platte des Wirts
  - Crypto-Versiegelung duch Checksumme
  - beim Einlagern: Checksummenvergleich, Entschlüsselung
  - $\oplus$ auch bei wenig internem Speicher zukunftsfähig
  - $\ominus$ Geschwindigkeit, CPU-Belastung

Anwendungen für sichere Koprozessoren: Softwarekopierschutz

- Verkaufsmodelle
  - per-CPU
  - per-site
  - per-use

- Dongles
  - ⊖ nur für eine Applikation
  - ⊖ "Schlange" am Druckerport oder USB $\leadsto$ Verträglichkeitsprobleme
  - ⊖ nicht geeignet für "per-use" oder Verleih

Anwendungen für sichere Koprozessoren: Softwarekopierschutz[2]

- CPU-IDs
  - ⊖ Einkompilieren in Applikation $\rightsquigarrow$ Wartung?
  - ⊖ Rechnerwechsel?
  - ⊖ Reverse Engineering
- Internet
  - ⊖ unsicheres Medium
  - ⊖ Datenschutz
  - ⊖ firmenweite Firewalls/Sicherheitsrestriktionen

Anwendungen für sichere Koprozessoren: Softwarekopierschutz[2]: sicherer Koprozessor

- Sicherer Koprozessor speichert Schlüssel
- “Multifunktions-Dongle”
- Verschlüsselte Binaries erlauben Backups
- speziell zugeschnittene Software
  - läuft komplett oder teilweise am

  - $\rightarrow$ Crypto-Paging

- Voraussetzung: ungehacktes OS $\rightarrow$ Schicht 0 ROM Bootstrap

Elektronische Währung

- Generierung/Eliminierung von Geld
- Eigenschaften (wie ACID bei RDBMS)
  - **Fehleratomarität** bei Transaktionsabbruch zurück zum Start
  - **Permanenz** korrekte Zustände bleiben erhalten
  - **Serialisierbarkeit** Reihenfolge bei Programmnebenläufigkeit egal
    $\leadsto$ selbes Endergebnis wie bei serieller Ausführung

Elektronische Währung: Geldanalogie

- Geldanalogie
  - Anonym
  - Seriennummern
  - Generierung/Eliminierung nur durch “Notenbanken”
  - sicherer Koprozessor von Käufer und Verkäufer
    * sicherer Kanal
    * atomare Transaktion
    * ACID Rollbackmechanismen

Elektronische Währung: Bank-Rendezvous-Analogie

- Bank-Rendezvous-Analogie
  - zentralisierter “Bank”-Server

  ⊕ einfache Zugriffsrechteverwaltung, Kontoverwaltung

  ⊖ schlechte Skalierbarkeit

  ⊖ zentrale Fehlerquelle/Agriffsziel (DoS-Attacken)

  ⊖ Verlust der Anonymität

Elektronische Währung: Kreditkarten/Scheckanalogie

- Kreditkarten/Scheckanalogie
  - Käufer und Verkäufer authentifizieren und garantieren Liquidität

  $\ominus$ für Liquiditätsprüfung im voraus $\rightarrow$ erst recht Kommunikation zu einem Bankserver notwendig

  $\leadsto$ zurück zu Geld- oder Bank-Rendezvous-Analogie

“Point-of-Sale” Terminals

- nicht vertrauenswürdiges Gerät
- keine Garantie für Datenpfade (Tastatur, Bildschirm)
- zB am Bildschirm: 10 ATS Batterien, intern: 1 Mio. ATS Golduhr
- portables Token mit sicherem Anzeigegerät pro Benutzer
- sicheres Anzeigegerät reicht auch für Passworteingabe:
  - sicherer Koprozessor zeigt an: “QZKNCFLXW” (zufallsverschlüsseltes Passwort)[9]
  - mit Pfeiltasten rauf/runter/rechts zum ändern auf zB “ZAHNPASTE”

Alternativen

- DigiCash ([10] und [11])
- NetBill ([12])
- reine Softwarelösungen
- verlassen sich auf heutige Sicherheitsstandards im Internet
- $\rightsquigarrow$ unsichere Schlüsselverwaltung
- $\rightsquigarrow$ und/oder Zentralisierung

Elektronische Verträge

- multi-party Verträge: elektronische Marktplätze, “Superdistribution”
  - Käufer darf im Namen des Erzeugers wieder weiterverkaufen
  - Basispreis darf nicht unterschritten werden
  - Wiederverkäufer kann das Produkt nicht verändern
- Flugbörsen
  - Kundenanfrage: max. Preis, Ziel, Datum
  - Reiseagenturen: bieten mit
  - bei Zuschlag: kauft Karten bei Fluggesellschaft
  - Karten sind auch verschlüsselte Objekte
  - alle Transfers über sichere, ACID-konforme Kanäle

Grundsätzliche Sicht

- alle geschützten Objekte
  - in sichere Koprozessoren
  - ACID Transaktionsgarantien
  - korrekter Authentifizierungsschutz
- für Verhalten: ausdrucksstarke Beschreibungssprache
  - Authentifizierung
  - Objekttransfer zu anderen Kommunikationspartnern
  - Zeitrestriktionen (zB Öffnungszeiten und Deadlines)
  - grundsätzliche Programmiersprachenprimitiven

Sichere Post

- schützenswerte Teile
  - Schutz von Briefinhalt
  - Verschlüsselung von Sender/Empfänger
  - Briefmarken
- U.S. Postal Services
  - 40.000 Postämter
  - 165 Milliarden Poststücke jährlich([13])
  - Frankiermaschine zieht Markenwert intern ab

Sichere Post[2]

- Vier mögliche Attacken
  - modifizierter Frankierzähler
  - gefälschte oder kopierte Briefmarken
  - unerlaubte Benutzung der Frankiermaschine
  - Diebstahl einer Frankiermaschine (82.000 verloren oder gestohlen!!!)

Sichere Post[3]: Digitale Crypto-Marken

- Digitale Crypto-Marken
  - Barcode oder PDF417-Codierung ([14])
  - PC löst Authentifizierungsproblem
  - drucken mit normalem Laserdrucker
  - wichtige Information am Poststück
    - Absender- und Empfängeradresse
    - Wert der Marke
    - Serien# und Sequenz# der Softwareinstanz (im Postamt)
    - Zeitstempel
  - Unfälschbar
  - Kopien gehen immer an gleichen Empfänger

Sichere Post[3]: Erkennen von unerlaubten Kopien

- erste Hochrechnung
  - ein Kilobyte pro Marke
  - 165 Milliarden Marken
  - $\leadsto$ erscheint komplett sinnlos
- natürliche physische Verteilung von Empfängern
  - Sortierung nach PLZ
  - 600 regionale Vorverteilerposten
  - max. 6 Monate “Lebensdauer” einer Marke
  - 80 Mrd. Inlandsversendungen
  - $\leadsto$ 130 Mio. Marken $\rightarrow$ unter 130GB

Sichere Post[3]: Erkennen von unerlaubten Kopien: Optimierungen

- Optimierung
  - Check beim Eingangspostamt
    - hohe Wahrscheinlichkeit, dass beide Poststücke im gleichen Gebiet aufgegeben werden
  - anderer Aufgabeort
    - Standleitungen zwischen Vorverteilerposten
    - Vergleiche on-line oder im Batch-Betrieb
    - Dimensionierung abhängig notwendiger Aufdeckungsgeschwindigkeit

Zusammenfassung

- Rückblick
  - Gefahren und deren Beseitigung durch sichere Koprozessoren
  - Familie der sicheren Koprozessoren
  - Hardware- und Softwareanforderungen
  - zwei Beispielimplementierungen
  - High-Level Applikationen für sichere Koprozessoren
- Abschluss
  - fehlende Benutzerakzeptanz
  - versteckte Sicherheit durch Standardgeräte in PCs
  - adaptierte Standardanwendungen (Email clients, Dateisysteme...)

Literatur

[1] Bennet Yee, J.D. Tygar, “Secure Coprocessors in Electronic Commerce Applications”

[2] “Dyad” Homepage: http://www.cs.cmu.deu/afs/cs/project/dyad/www/

[3] Joan Dyer, Ron Perez, Sean Smith, Mark Lindemann, “Application Support Architecture for a High-Performance, Programmable Secure Coprocessor”

[4] Elaine R. Palmer, Sean W. Smith, Steve Weingart, “Using a High-Performance, Programmable Secure Coprocessor”

[5] Sean W. Smith, “Secure Coprocessing Applications and Research Issues”

[6] Richard M. Karp and Michael O. Rabin, “Efficient randomized pattern-matching algorithms” Technical Report TR-31-81, Aiken Laboratory, Harvard University, December 1981

[7] Blum, Blum, and Shub, “Comparison of two pseudo-random number generators”, Advances in cryptology: CRYPTO-82, pages 61-79, 1983

[8] Namuel Blum and Silvio Micali, “How to generate cryptographically strong sequences of pseudo-random bits”, SIAM Journal on Computing, 13(4):850-864, November 1984

[9] M. Abadi, M. Burrows, C. Kaufman, and B. Lampson, “Authentication and delegation with smart-cards”, Technical Report 67, DEC Systems Research Center, October 1990

[10] Stefan Brand, “An efficient off-line electronic cash system based on the

representation problem", Technical Report CS-R9323, Centrum voor Wiskunde en Informatica, 1993

[11] David Chaum; "Security without identification: Transaction systems to make big brother obsolete", Communications of the ACM, 28(10):1030-1044, October 1985

[12] Marvin Sirbu and Doug Tygar, "Netbill: An internet commerce system optimized for networked delivered services", IEEE Compcon '95 Coference, pages 20-25, March 1995

[13] U.S. Postal Service, "Annual report of the postmaster general", fiscal year 1991

[14] Stuart Itkin and Josephine Martell, "A PDF417 primer: A guide to understanding second generation bar codes and portable data files",

Technical Report TR-31-81, Aiken Laboratory, Harvard University, December 1981