# Einführung in verdeckte Kanäle ("Covert Channels")

9. Augsburger Linux-Infotag
27. März 2010

Steffen Wendzel

www.wendzel.de / www.linux-openbook.de

# Agenda

- Hauptsächlich eine Einführung
- Teilweise kurzer Ausblick auf fortgeschrittene Themen
- Einführung → Techniken → Anti-CC-Methoden

# I. Einführung

- Zum Begriff
- Confinement Problem
- Bell-LaPadula-Modell
- Prisoners' Problem

# Zum Begriff

- „*Covert Channels, i.e. those not intended for information transfer at all, such as the service programm's effect on the system load.*“

  – B. Lampson

- „*A covert channel is a path of communication that was not designed to be used for communication.*“

  – M. Bishop

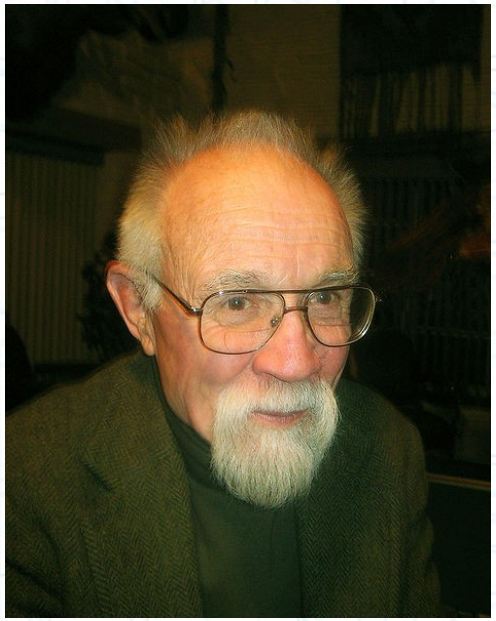
- Weitere Begriffe (nicht immer exakt gleiche Bedeutung in akademischer Welt):

  Hidden Channels, Steganographic Channels, Sublimminal Channels, Tunneling (Hacker-Community), Side Channels

# Anfänge

- 1973: Butler Lampson „A Note on the Confinement Problem“
- 1973: Bell-LaPadula-Modell
  - NWD, NRU
- 1983: Gustavus J. Simmons: „The Prisoners' Problem and the Subliminal Channel“

Lampson, (Quelle: eecsfacweb.mit.edu)



Simmons (Quelle: Wikipedia)

# Confinement Problem von Butler Lampson

- Customer-Prozess baut Hülle um Service-Prozess
- Confinement Problem: Service-Prozess soll keine vertrauenswürdigen (confidential) Daten vom Customer an unberechtigte weitergeben.
- Stellt drei Channel-Typen vor, darunter:

  Covert Channel […] „*those not intended for information transfer at all, such as the service programm's effect on the system load.*“

# Prisoner's Problem von Gustavus J. Simmons

- Nicht „Prisoner's dilemma“ (→Spieltheorie)!
- Zwei Gefangene (getrennte Zellen)
- Kommunikation nur über Wärter
- Wärter kann jegliche Kommunikation unterdrücken, verändern, selbst erstellen

→ „Subliminial Channel“

In akademischer Welt Covert Channel != Subliminal Channel → Moskowitz et. al.

# Kurz und Grob: Storage und Timing Channel

- Covert Storage Channels
  - Verändern Attribute (etwa Dateigröße oder Dateinamen)
- Covert Timing Channels
  - Verändern Reihenfolge (etwa von Netzwerkpaketen → Ahsan: „Covert Channel Analysis and Data Hiding in TCP/IP“, 2002) oder Zeitwerte (Programm wartet *num* Sekunden).

# Warum wichtig?

- X verschlüsselt Daten mit einem tollen Verfahren in Land Y
- Die Kommunikation von X wurde überwacht und am nächsten Tag steht die Geheimpolizei vor seiner Tür.

→ X gibt die verschlüsselten Informationen frei

Covert Channels sind wichtig für freie Meinungen und freies Wissen!

# Wer nutzt Covert Channels?

- Darüber kann man nicht viel sagen!
- vielleicht (oder: wahrscheinlich)
  Geheimdienste
- Botnetze

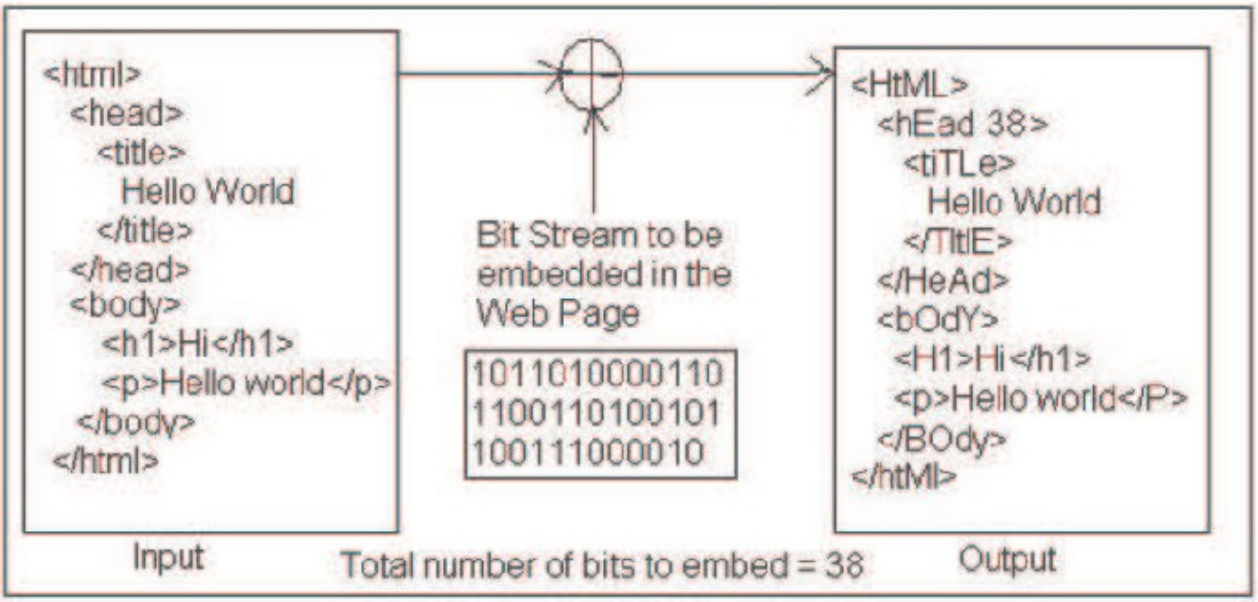
# II. Techniken zur Erstellung von Covert Channels

- Klassische Covert Channels
- Covert Channels in TCP/IP
- Fortgeschrittene Techniken

# Wie kann man verdeckt kommunizieren?

- Generell bietet die Steganografie dafür unzählige Möglichkeiten

→ s. Buch von K. Schmeh, „Versteckte Botschaften“, Heise-Verlag, 2009



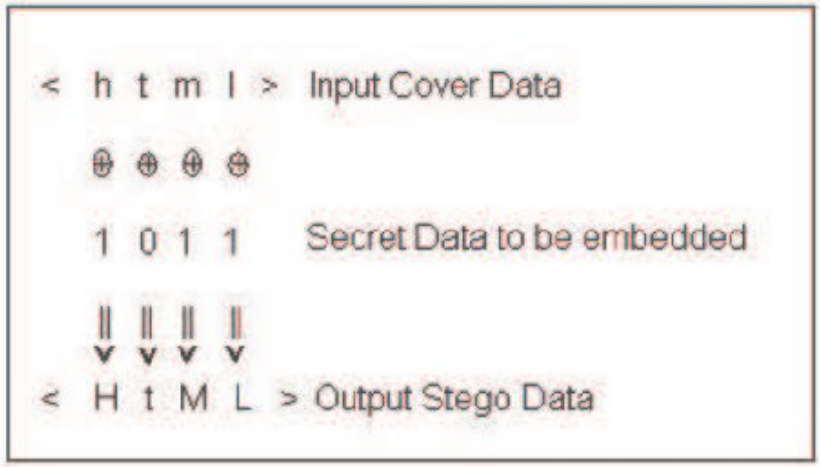




**Fig. 1.** Illustration of how the Html data hiding works

Quelle: Al-Qaheri et al.: Hiding inside HTML and Other Source Codes.

# Wie kann man verdeckt kommunizieren?

- Generell bietet die Steganografie dafür unzählige Möglichkeiten

  → s. Buch von K. Schmeh, „Versteckte Botschaften“, Heise-Verlag, 2009

- CC als Unterthema der Steganografie bieten unzählige weitere Möglichkeiten

  → mehr dazu auf den nächsten Folien

# Klassische Covert Channels

- Datei vorhanden/nicht vorhanden → simples Protokoll (das Standardbeispiel)
- CPU-Auslastung (ebenfalls Standardbeispiel)
- Typischer Request-Response-Timing Channel (→ Eßer: Ausnutzung verdeckter Kanäle am Beispiel eines Web-Servers, Diplomarbeit, 2005).

# Covert Channels in TCP/IP

- Beispiel HTTP
  - User-Agent (Version), HTTP-Version, URL, Host, …
- SMTP und NNTP
  - XHDR/XOVER range, XHDR type, Groß-/Kleinschreibung, Header-Bestandteile, Newsgroup-Namen, ...
- TCP
  - ISN, TCP-Flags, Urgent-Pointer, ...

# Covert Channels in TCP/IP

- IPv4
  - (Nicht-)Verwendung von Optionen,
  - Fragmentierung, Identification-Bits, uvm.
- ICMPv4
  - Echo Request/Response Payload
  - ICMP Address Mask Request und weitere
- IPv6 (inklusive IPSec)
  - Über 20 Möglichkeiten bekannt

    (→ Lucena et. al.: „Covert Channels in IPv6“)

# Fortgeschrittene Techniken

- Protokollwechsel
  - Statisch LOKI2 → daemon9, 1997
  - Randomisiert (phcct) → Wendzel, 2007
- Protocol Channels → Wendzel, 2009
- Dynamische Protokoll-Absprachen (Paper von Yarochkin, Dai et. al.: *Towards Adaptive Covert Communication System*, 2008)
- Passive Covert Channels (PCC)

  → NUSHU (Rutkowska, 2004)

# Freie Tools für CC/Tunneling

- LOKI2 (DNS, ICMP), phcct (PO3, HTTP, FTP), pct (ARP, ICMP), vstt (ICMP, POP3, ...)
- Httptunnel (HTTP)
- Pingtunnel (ICMP)
- simple icmp tunnel (ICMP)
- NUSHU (Passive Covert Channel)
- Stegtunnel (IP ID, SeqNr)
- Corkscrew (SSH über HTTP)

# III. Erkennung und Vermeidung von Covert Channels

- Shared Resource Matrix
- Covert Flow Trees
- Pump

# Shared Resource Matrix

- Kemmerer '83
- Vorgehensweise:
  1. Überprüfen, welche Ressourcen (und Attribute) sichtbar sind.
  2. Operationen feststellen, die auf diese Attribute (lesend/schreibend) zugreifen.

     Etwa (Standardbeispiel aus Bishop, vereinfacht):

| | read() | create() | delete() |
|---|---|---|---|
| Existenz | R | R,M | R,M |
| Label | R | M | R |

# Shared Resource Matrix

- Forts. Beispiel:
  - High Prozess → Low Prozess
  - High erstellt (create()) bzw. löscht (delete()) eine Datei
  - Low benutzt create() → Label von Low nicht wichtig, da Low Fehler registrieren wird, falls Datei bereits existiert.

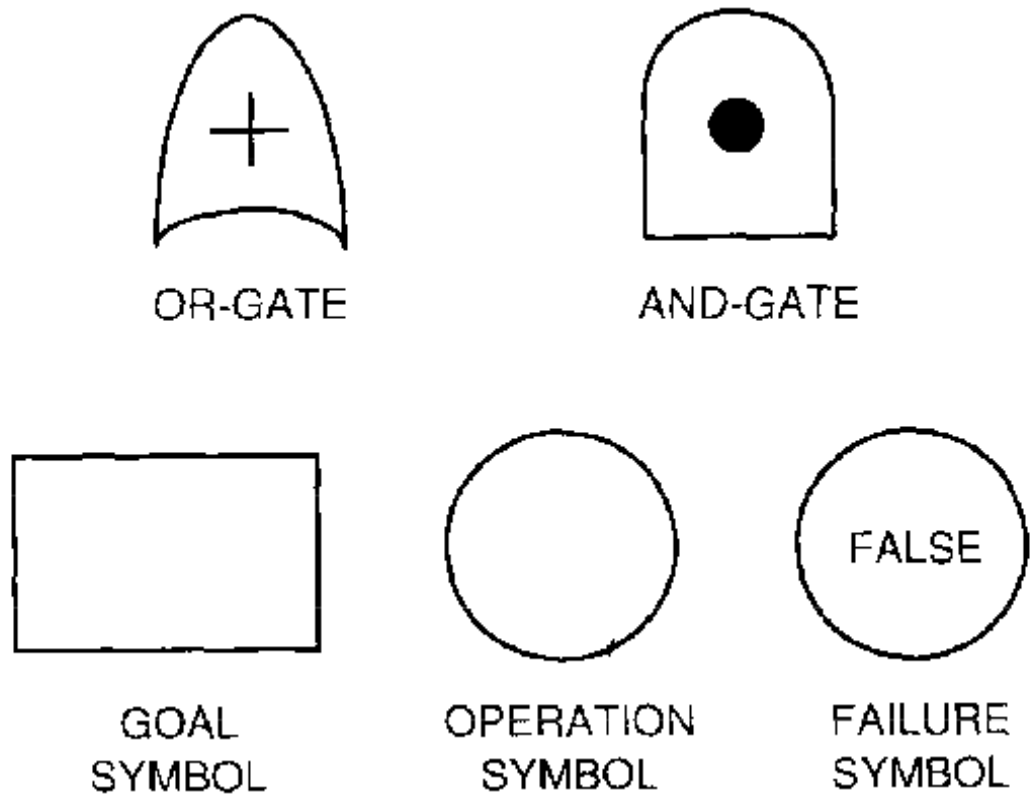
| | read() | create() | delete() |
|---|---|---|---|
| Existenz | R | R,M | R,M |
| Label | R | M | R |

# Covert Flow Trees

- R. Kemmerer und P. Porras: „*Covert Flow Trees: A Visual Approach to Analyzing Covert Storage Channels*“, IEEE Tr. on SE, Vol. 17, No. 11, Nov. 1991
- Verfahren zur Erkennung von Covert Storage Channels
- Basiert auf Fehlerbaumanalyse (*fault tree analysis*)
- Beispiel basiert auf M. Bishop: „*Computer Security. Art and Science*“, AW, 2003.

# Covert Flow Trees

- Verschiedene Knoten für Baum-Aufbau:
- Im Folgenden ein gekürztes Beispiel aus M. Bishop (ohne indirekte Covert Storage Channel-Erkennung)




Quelle: Paper von Kemmerer/Porras

# Covert Flow Trees

Schritt 1: Code überprüfen und Verwendung/Modifikation/Rückgabe von Attributen erkennen.

```
1  (* Datei locken, wenn noch nicht geöffnet u. gelockt *)
2  procedure Lockfile(f: file);
3  begin
4     if not f.locked and empty(f.inuse) then
5        f.locked := true
6  end;
7
8  (* Datei unlocken *)
9  procedure Unlockfile(f: file);
10 begin
11    if f.locked then
12       f.locked := false
13 end;
14
15 (* Prüfen, ob Datei gelockt ist *)
16 function Filelocked(f: file): boolean;
17 begin
18    Filelocked := f.locked;
19 end;
```

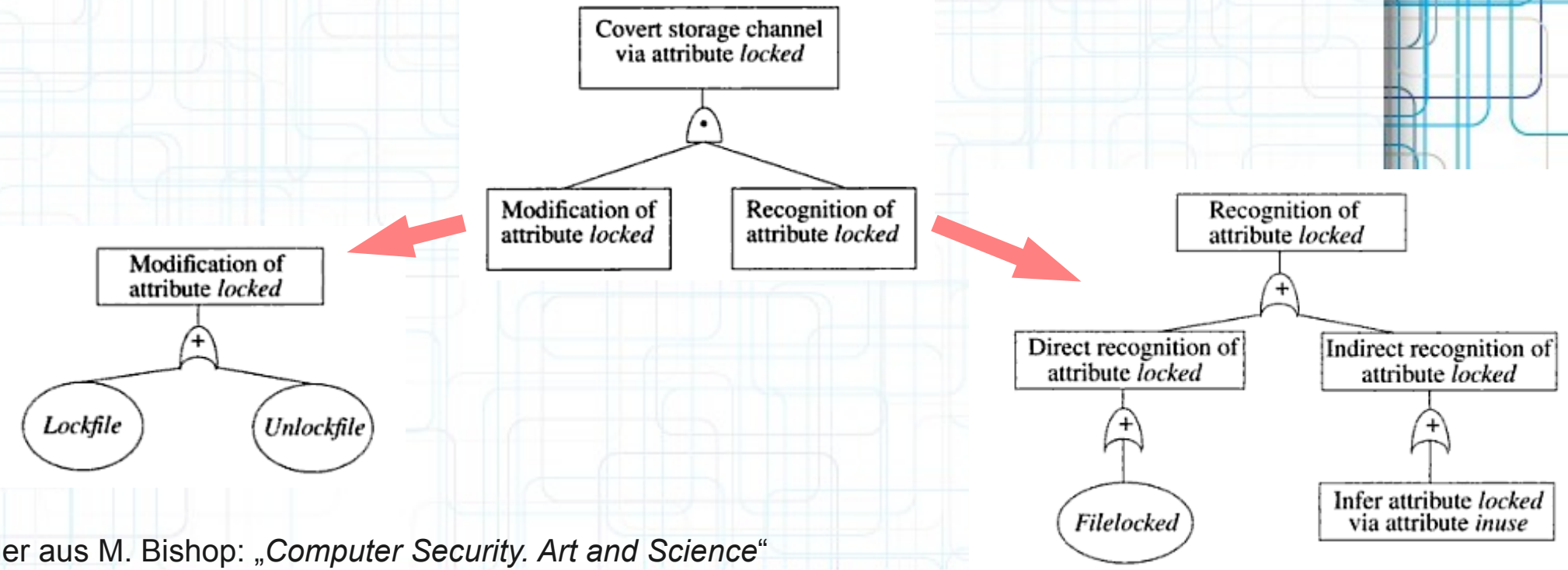
| | Lockfile | Unlockfile | Filelocked |
|---|---|---|---|
| reference | locked,inuse | locked | locked |
| modify | locked | locked | - |
| return | - | - | locked |

Beispiel aus M. Bishop: „*Computer Security. Art and Science*“ (modifiziert und gekürzt!)

# Covert Flow Trees

Schritt 2: Covert Storage Channel finden und Covert Flow Tree aufbauen (komplexer Schritt, daher nur in absoluter Kurzform).

| | Lockfile | Unlockfile | Filelocked |
|---|---|---|---|
| reference | locked,inuse | locked | locked |
| modify | locked | locked | - |
| return | - | - | locked |




Bilder aus M. Bishop: „*Computer Security. Art and Science*“

# Covert Flow Trees

- Information Flow Listen erstellen
  - 1. Liste: Operationsfolgen, die Attribute modifizieren
  - 2. Liste: Operationssequenzen, die Modifizierungen prüfen können

  List 1 = ( ( Lockfile ), ( Unlockfile ) )

  List 2 = ( ( Filelocked ) )
- Damit ist der Informationsfluss für den Covert Storage Channel:
- Lockfile → Filelocked bzw. Unlockfile → Filelocked

# Die Pump

- Kang u. Moskowitz '93
- Daten von Low zu High über Pump (Zwischenpufferung)
- Einweg-Kommunikation (mit ACKs):

  LOW ---DATA-->PUMP--DATA-->HIGH

  LOW <---ACK----PUMP<---ACK---HIGH
- Ziel: Gefahr eines Covert Channels durch ACKs senken
- Basic vs. Network Pump (→ mehrere Sender/Empfänger gleichzeitig)

# Ende

- Slides in Kürze auf meiner Webseite
  www.wendzel.de
- Dort gibt es auch hin und wieder aktuelle Informationen zum CC-Buchprojekt.
- Covert Channel Mailinglist (englisch)
  http://www.wendzel.de/?sub=showpost&blogid=1&postid=183

# Linux-Users-Welcome.de

LINUXUSERSWELCOME


Startseite | Material | Tipps | Projektnews | Erfolgsgeschichten | Impressum

Hier findet ihr Geschäfte, in denen es besonders angenehm ist,
Hardware für euer Linux System zu kaufen.

- In jedem dieser Geschäfte könnt ihr Hardware vor Ort mit eurem Linux Notebook oder einer mitgebrachten Linux Live-CD testen. Die Verkäufer unterstützen euch bei der Hardware Auswahl für Linux und helfen euch beim Testen der Hardware.
- Wenn sich ein Gerät nicht vor Ort testen lässt, dürft ihr es innerhalb einer Woche zurückgeben, sollte es nicht mit dem Linux eurer Wahl zusammenarbeiten.

**Verlosung von 3. Aufl. von „Linux. Das umfassende Handbuch“ → gibt’s auch in HTML-Form auf www.linux-openbook.de.**
**Zudem: 3 Jahres-Abos von Linux User!**