Jürg Gutknecht, SI und ETH Zürich, April 2015

Native Design: Ein Weg zu angriffssicheren Systemen?

Warum vertrauen wir hier? …

Der Staubsauger könnte ein Mikrofon eingebaut haben, welches sämtliche Geräusche im Raum aufnimmt und via Stromkabel an einen Geheimdienst weiterleitet

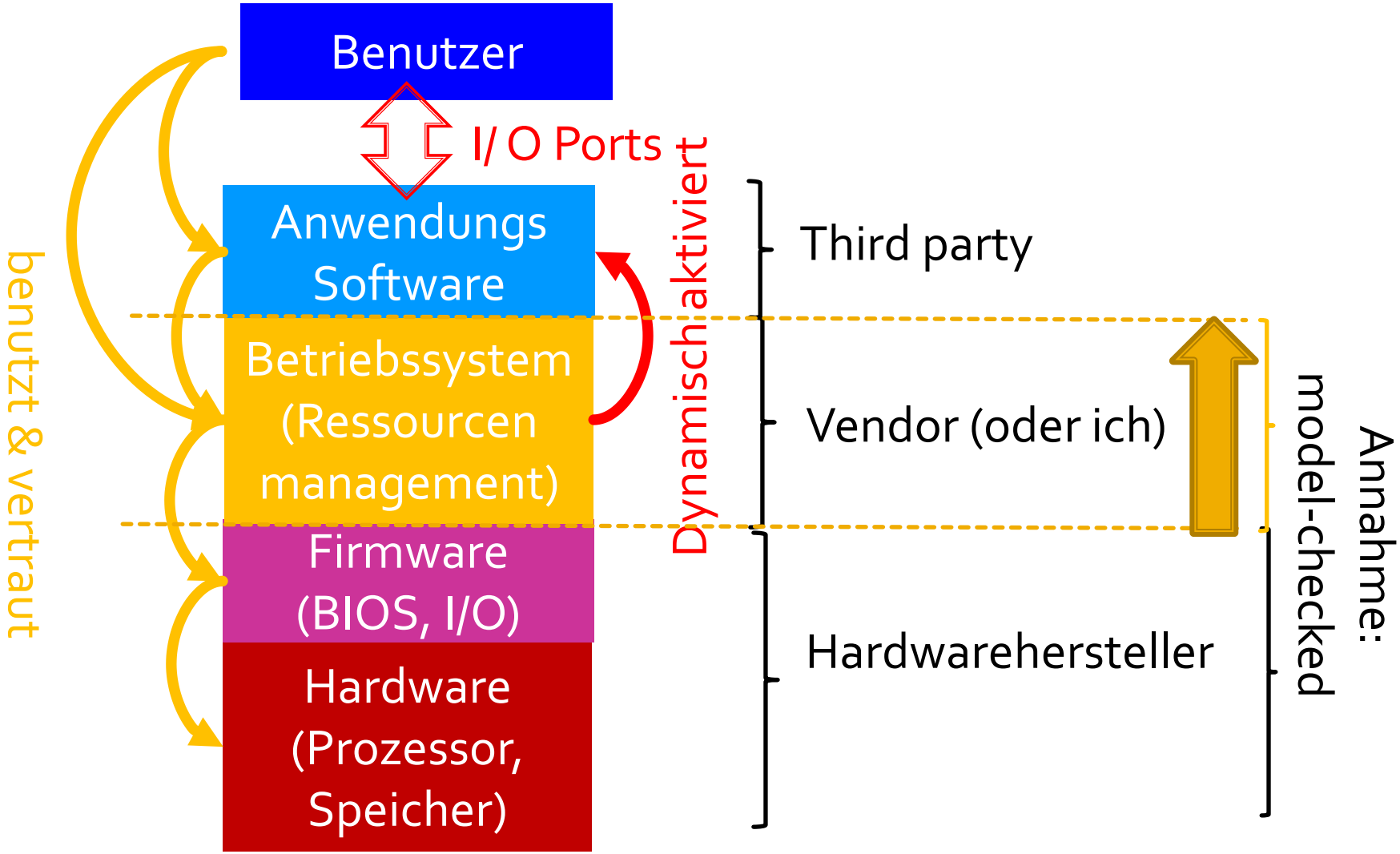
… und hier nicht?

Die Tastatur und das USB Verbindungskabel könnten manipuliert sein, so dass sie die gesamten Datenströme an den Geheimdienst weiterleiten

Mögliche Gründe

- Grösseres Vertrauen in Hersteller konventioneller Geräte
- Wesentlich höhere Komplexität der Computerhardware (und Firmware!)
- Konventionelle Geräte sind zu einem wohldefinierten Zweck *statisch* verdrahtet, während Computer Universalmaschinen sind, die ihr Verhalten *dynamisch* durch Einspeisen und Laden neuer Software ändern können

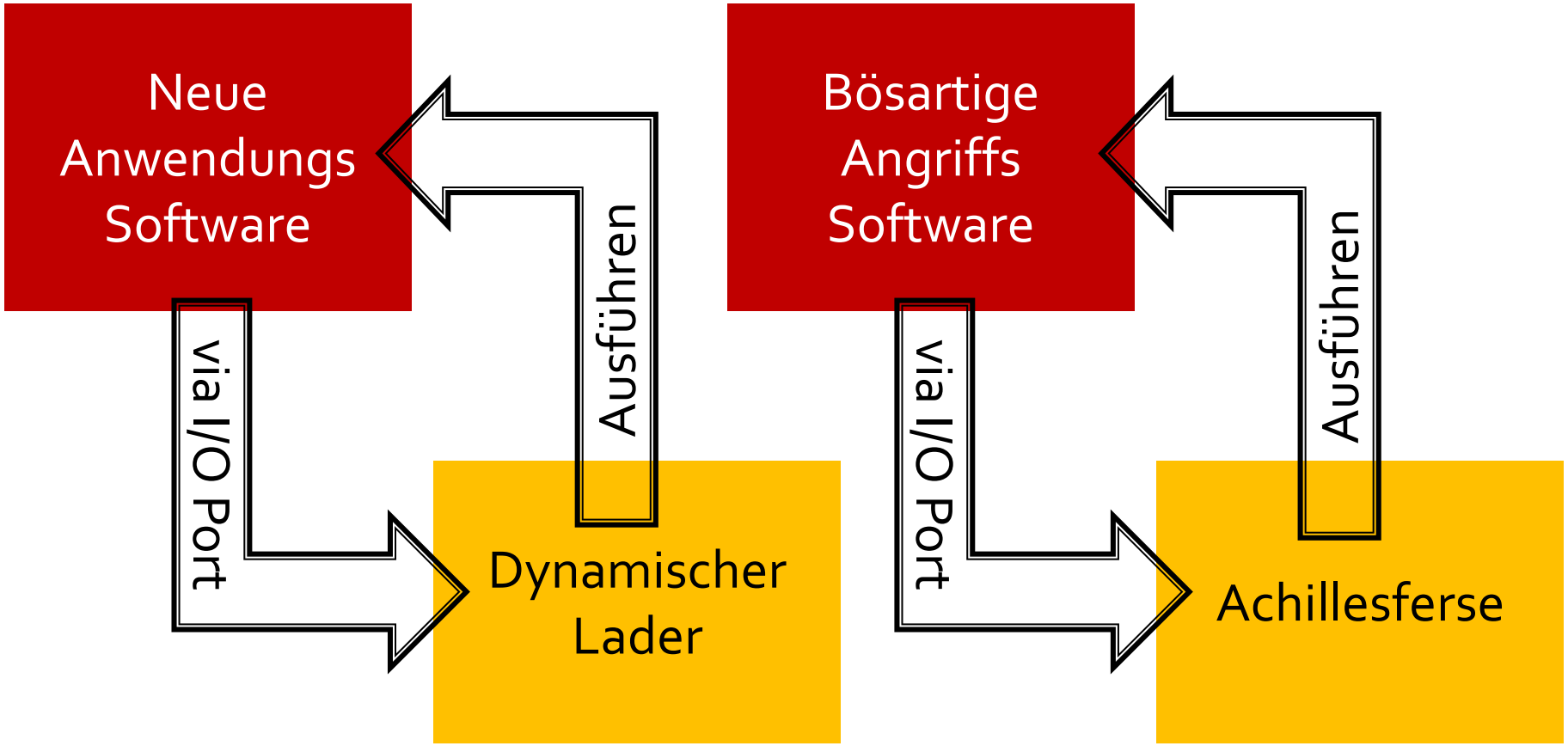
Vertrauensrelationen



Aufgaben des Betriebssystems

- …
- Speicherverwaltung
- Verwaltung von I/O Kanälen («Ports»)
- Ausführen von Kommunikationsprotokollen
- Aktivieren von Third Party Software
  - Dynamisches Laden und Ausführen von Code

Angriff auf Betriebssystem

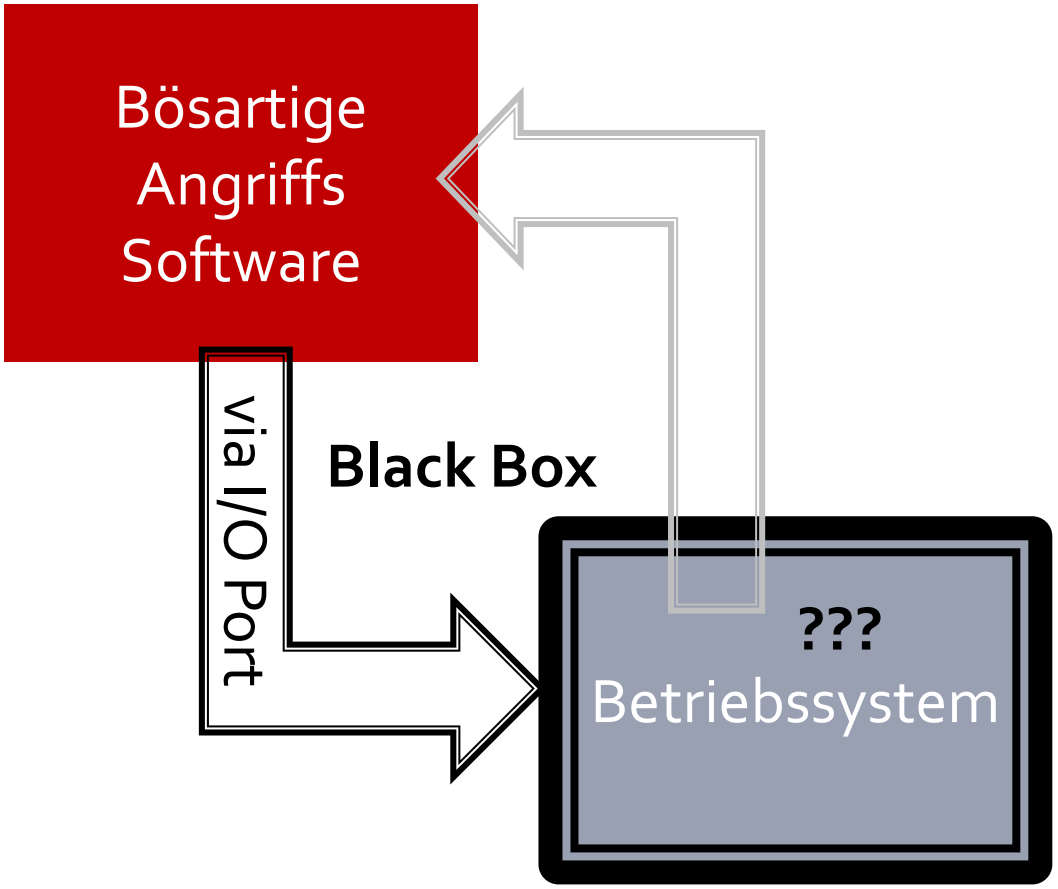


Achillesfersen

- Mangel in der Überprüfung der Kompatibilität von Eingabedaten mit Programmcode
- Existenz einer zufällig passenden Sequenz von Bytes im Speicher

- Basiert auf der Bekanntheit der Architektur des Betriebssystens
- Setzt Kenntnisse der Infrastrukur des Speichers voraus
- Funktioniert weil UNIX Kern und C verbreitet sind (Windows, Linux, OSX, …

Natives Betriebssystem



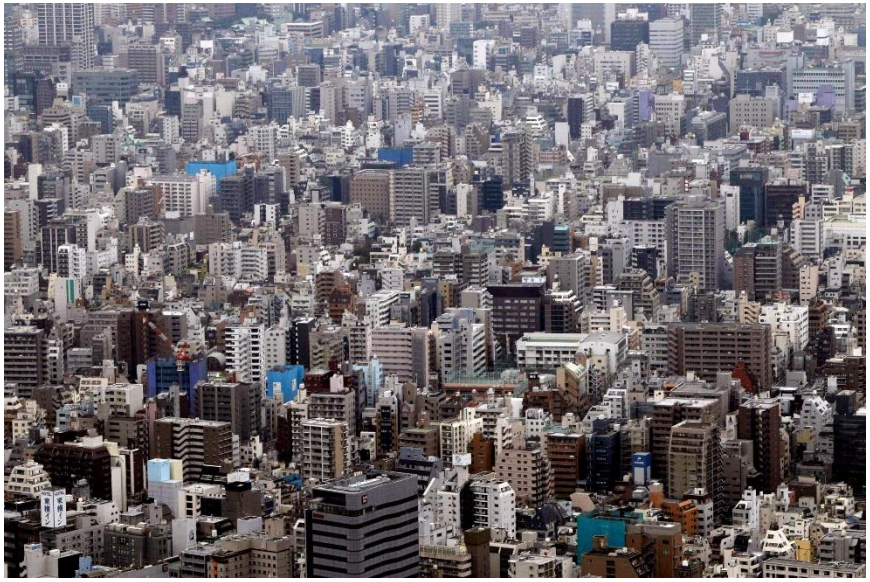
Oberon/ A2 (ETH Z)

- Natives Betriebssystem mit Möglichkeit zur Randomisierung der Infrastruktur
- Vollständige Kompatibilitätsprüfung
  - Programmiersprache „strongly typed“
- Ultrakompakt, minimalistisch, einfach
- Vollständig von einer Person überblickbar
  - Öffnung von Ports
  - Verwendung von Protokollen
  - Datenfluss von Eingabe bis Ausgabe

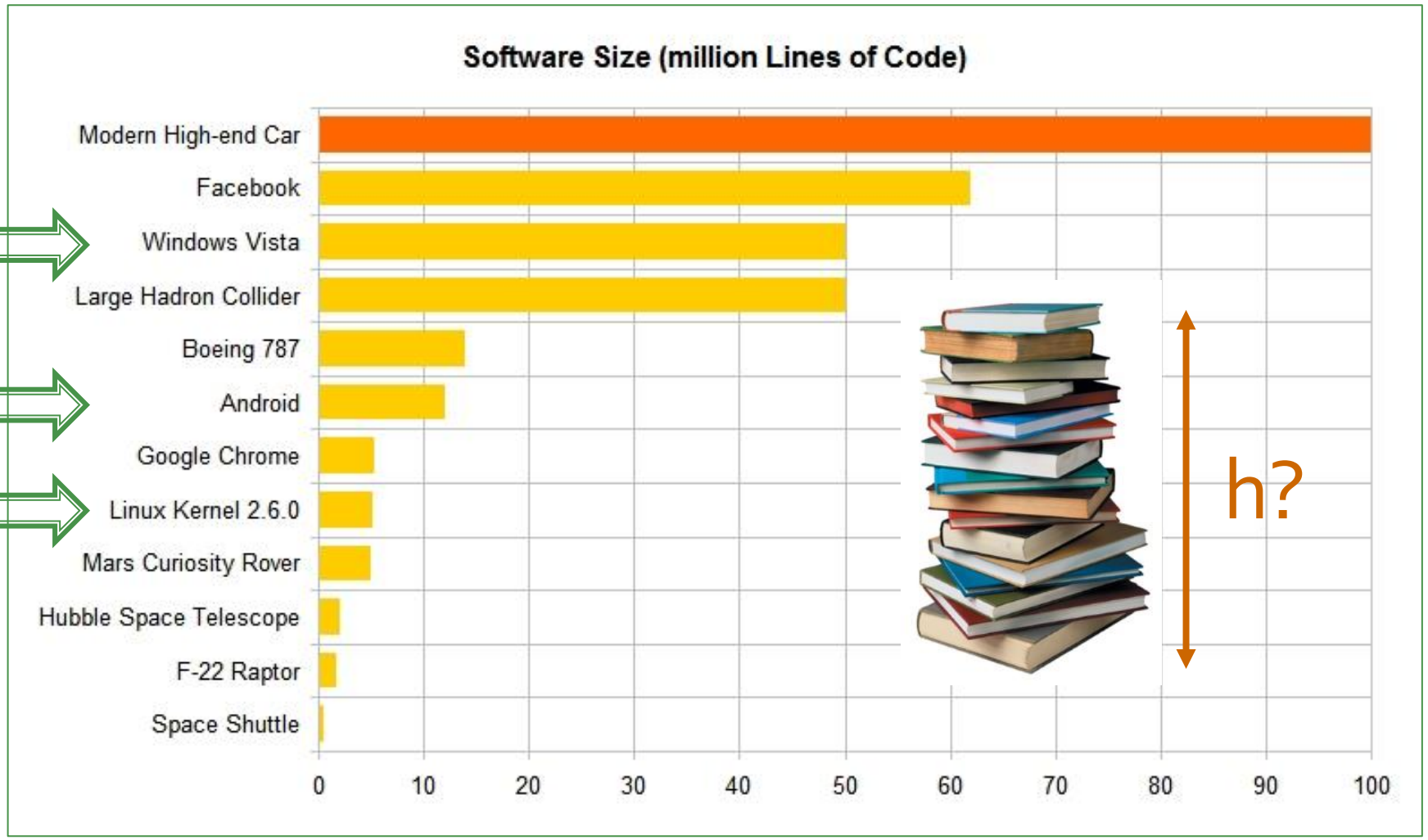
Village vs. Megacity

Oberon/ A2

Windows, Linux, OSX, ...



Grosse und kleine Systeme



Alles ist relativ…

- A2 Systemkerngrösse (Lockfreie Version)

Komponente	Codezeilen
Interrupt Handling	301
Memory Management (inkl. GC)	352
Modullader	82
Multiprocessing	213
Laufzeitunterstützung	250
Total	1198

Advocatus Diaboli

- Möglichkeit dass bereits die Herstellung des A2 Systems korrumpiert ist?
  - Alle Herstellungstools sind integraler Teil des A2 Systems selbst (Bootstrapping Konzept)

Universalität vs. Sicherheit

Universalität

Angriffssicherheit

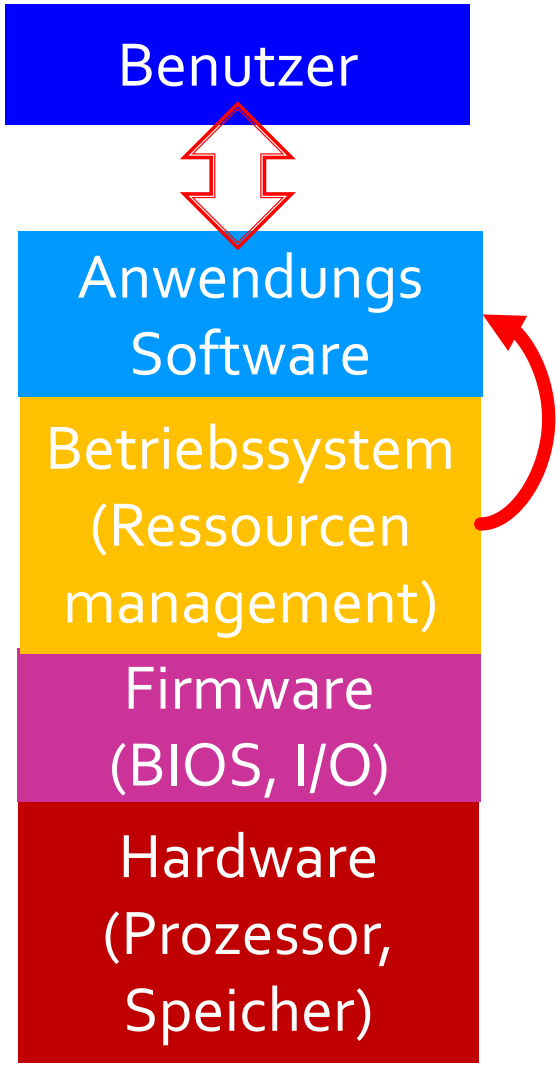
Maximale Universalität
Minimale Sicherheit

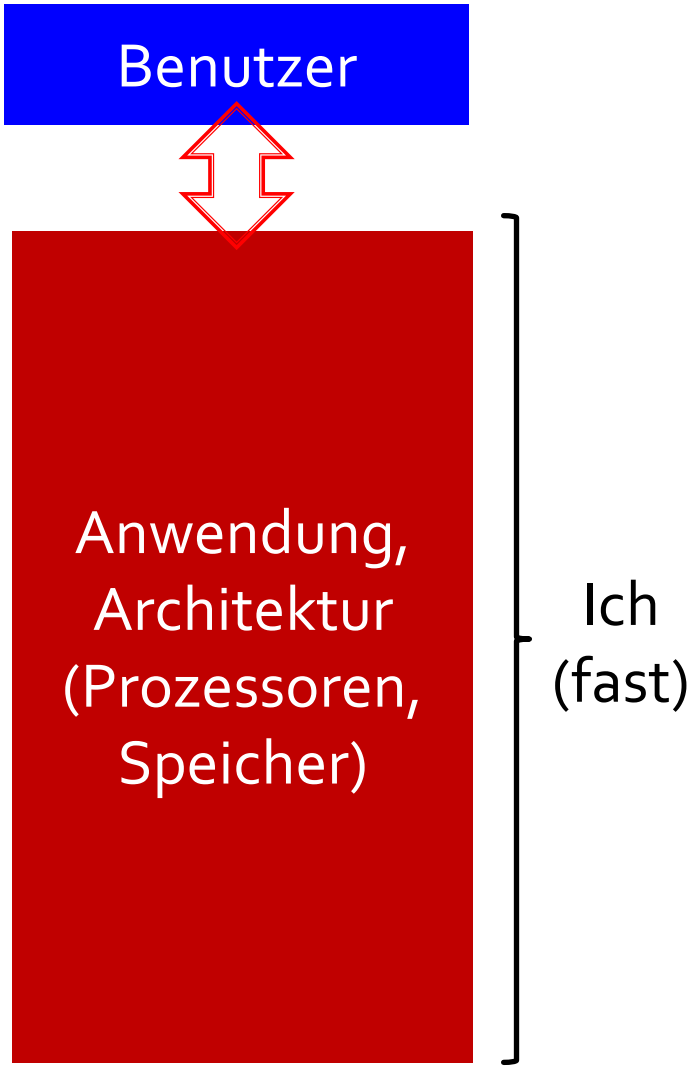
Minimale Universalität
Maximale Sicherheit

Security 2.0: Custom Design

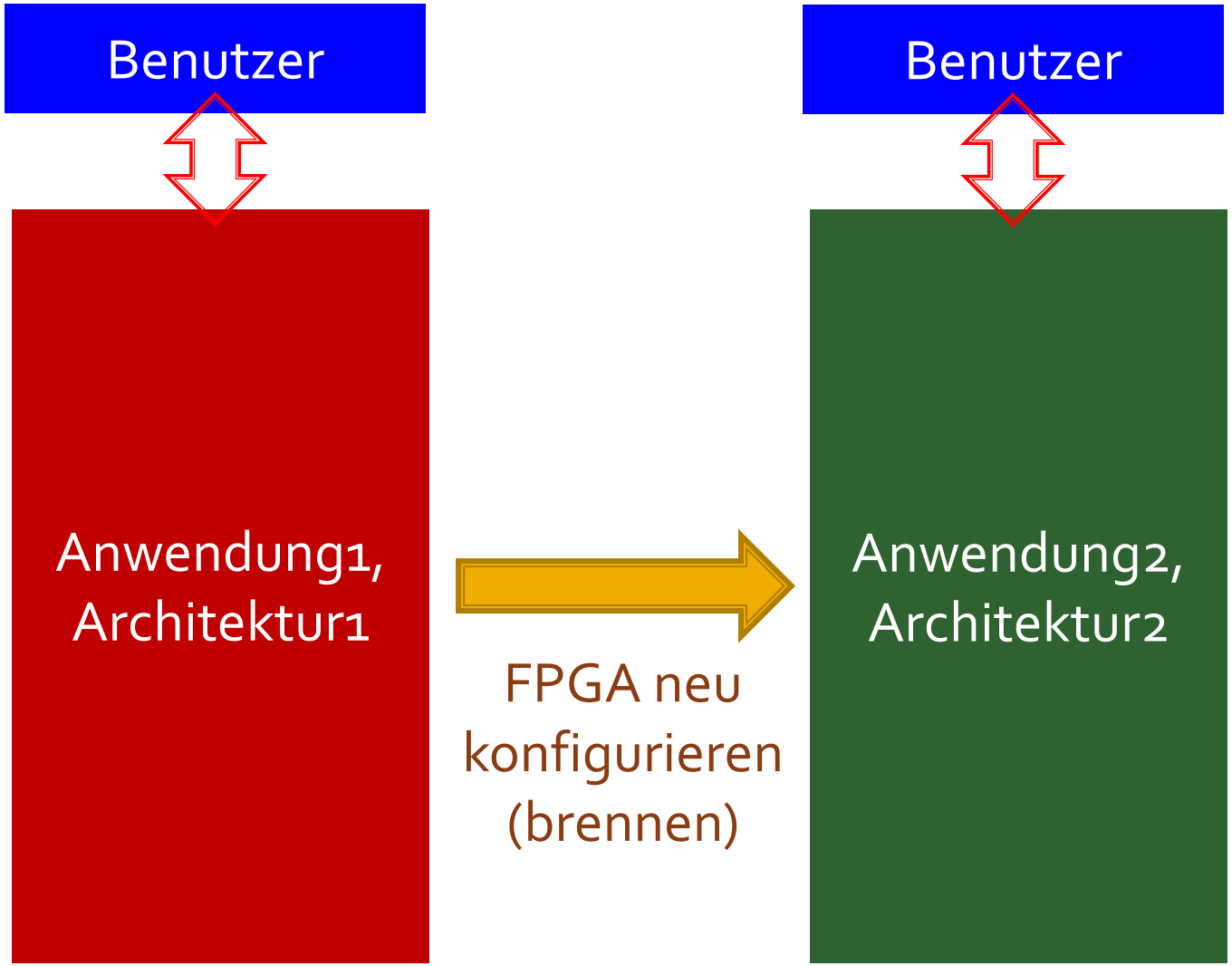
- Codesign Anwendung & Hardwarearchitektur
  - Basiert auf FPGA oder ASIC Technologie
- Auf spezifische Anwendung beschränkt
- Ressourceneffizient
- Konzeptbedingt crashsicher
- Attackensicher
- Konzept des Betriebssystems entfällt

Vertrauensrelationen 2.0





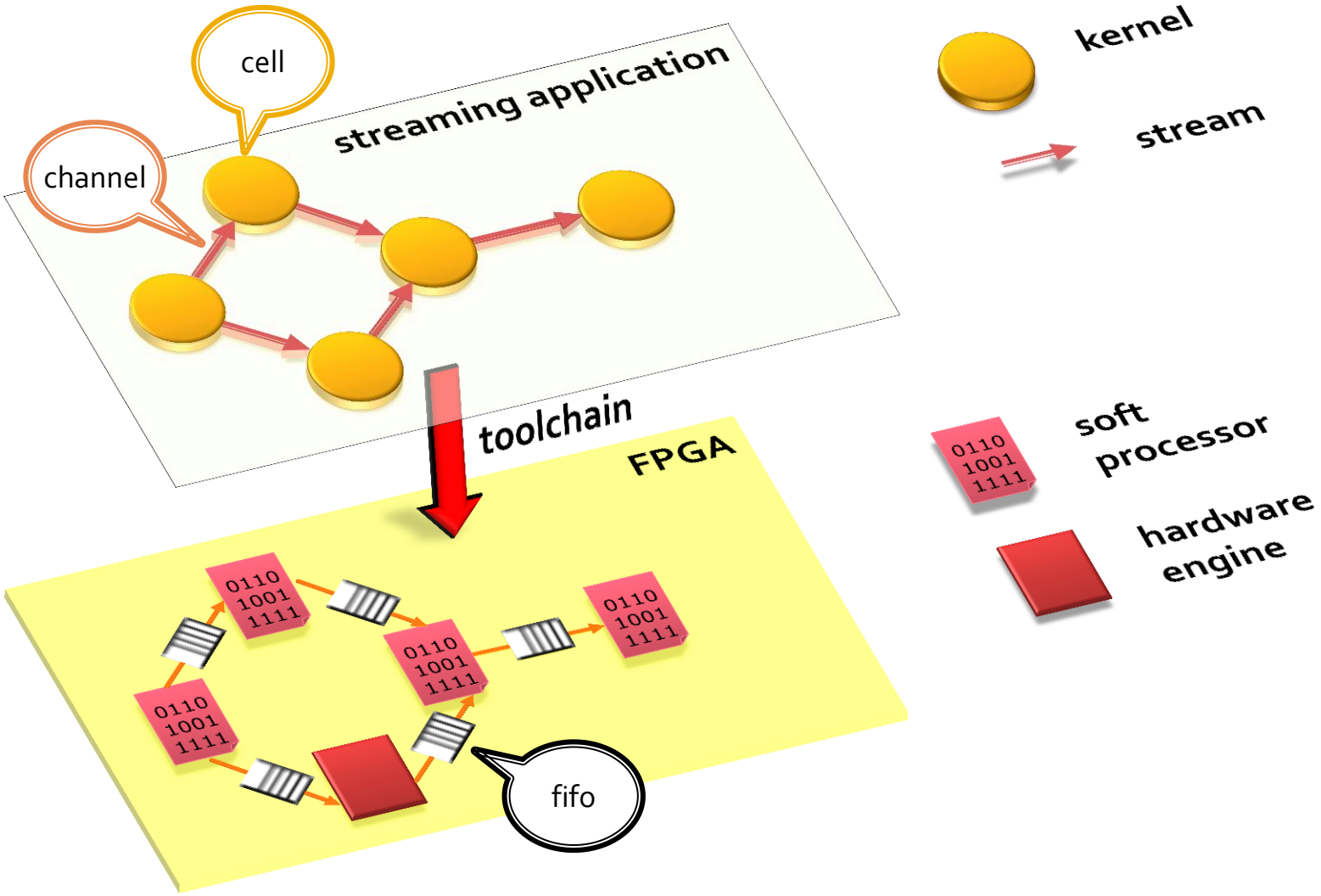
Konfigurierbare Hardware



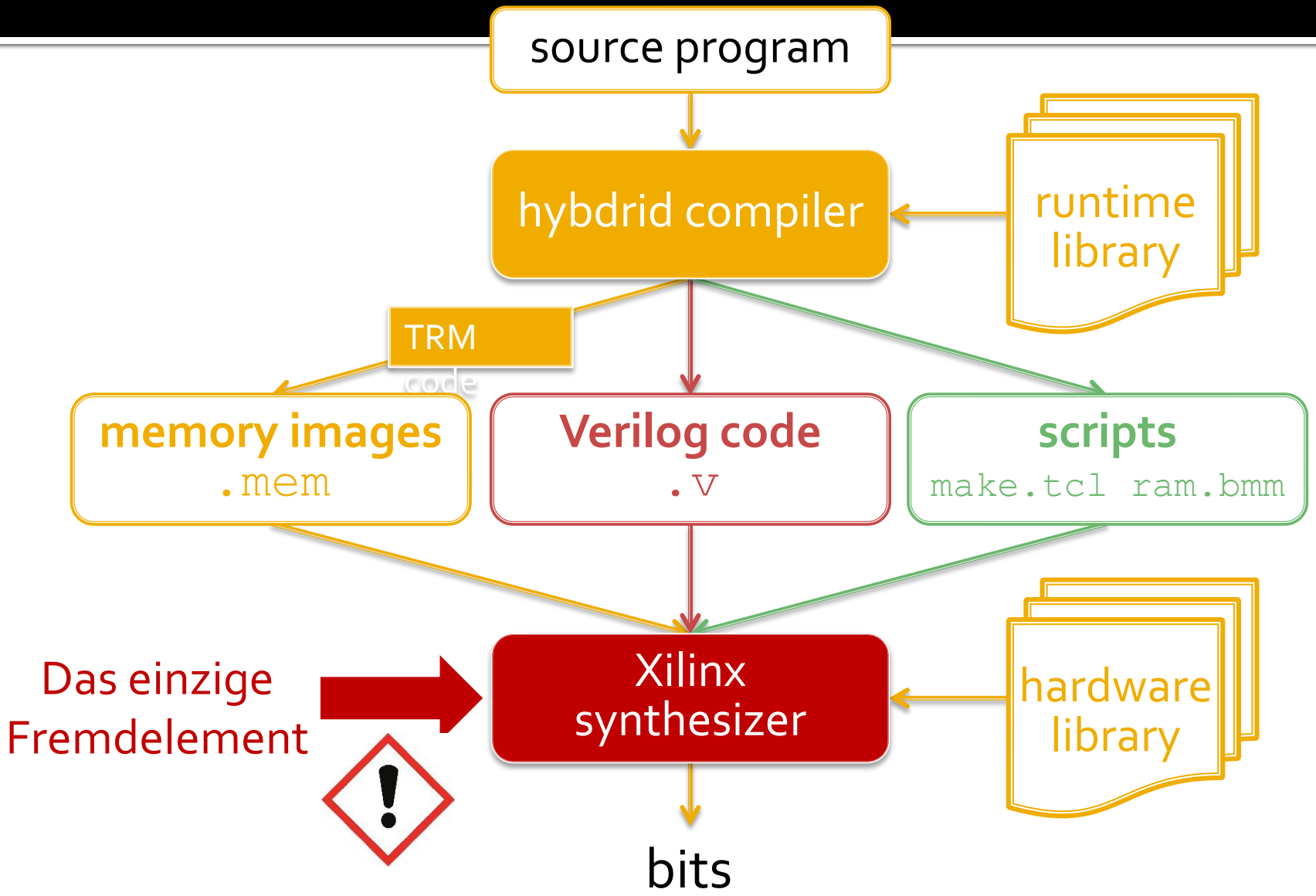
Das Active Cells Modell (ETH Z)

- Tool Chain für die holistische Entwicklung von IT Systemen auf hoher Abstraktionsebene
- Setzt keine Hardware Kenntnisse voraus
- Erzeugt hochparallele Manycore Designs
- Erlaubt heterogene Netzdesigns mit Knoten („Engines“) unterschiedlichster Art
  - Softcores
  - Optimierte Hardwarecores (Filter, Fourier, …)
- Setzt Vertrauen in FPGA Hersteller voraus

Entwicklung mit Active Cells



Tool Chain


source program
hybdrid compiler
runtime library
TRM
code
memory images
.mem
Verilog code
.v
scripts
make.tcl ram.bmm
Das einzige Fremdelement
Xilinx synthesizer
hardware library
bits

Warum vertrauen wir hier? …

Der Staubsauger könnte ein Mikrofon eingebaut haben, welches sämtliche Geräusche im Raum aufnimmt und via Stromkabel an einen Geheimdienst weiterleitet