A Secure and Reliable OS for Automotive Applications
Cars on Code

Matthias Lange
mlange {@} sec.t-labs.tu-berlin.de

TU Berlin
Spring 2010 SIDAR - Graduierten Workshop ber Reaktive Sicherheit

July 7th, 2010
Outline

1. Introduction
2. Architecture
3. Proof-of-Concept Implementation
4. Further Research
5. Important Related Work
Introduction
Introduction

Cars remained static for 80 years
- Gasoline engine
- Four wheels
- Familiar user interface

Change since late 70’s
- Computers coordinate and monitor sensors
- Ca. 100MB of binary code spread over 50 - 70 ECUs
Objective of Computerization

- **Safety**
  - Anti-lock Brake System
  - Airbag
  - Assistance

- **Value added features**
  - Automatic crash response
  - Remote diagnostics
  - Stolen vehicle recovery

- **Economically advantageous**
  - Cost
  - Weight
Future Trends

Consolidation

Classic control applications are supplemented by new complex applications on a single control unit.
Future Trends

Consolidation
Classic control applications are supplemented by new complex applications on a single control unit

Proactive Systems
Prevent or mitigate critical situations
Future Trends

Consolidation
Classic control applications are supplemented by new complex applications on a single control unit

Proactive Systems
Prevent or mitigate critical situations

User Customization
App Store, let the user buy “upgrades”
Challenges

Isolation

Assure spatial and timely isolation
Challenges

Isolation
Assure spatial and timely isolation

New Attack Vectors
Future trends will open a wide range of attack vectors for attackers.
Challenges

Isolation
Assure spatial and timely isolation

New Attack Vectors
Future trends will open a wide range of attack vectors for attackers.

New Threats
Computerized environments bring new array of potential new threats.
Architecture
Architecture

- L4 microkernel
  - Tasks, Threads and IPC
  - Object capabilities
Architecture

- L4 microkernel
  - Tasks, Threads and IPC
  - Object capabilities
- Infrastructure layer with basic services
  - Memory and IO manager
  - Secure GUI
  - Drivers
Architecture

- L4 microkernel
  - Tasks, Threads and IPC
  - Object capabilities
- Infrastructure layer with basic services
  - Memory and IO manager
  - Secure GUI
  - Drivers
- Reuse existing, collaboratively developed software
  - OS rehosting
  - Virtualization
OS Rehosting

- Adapt OS kernel to run as deprivileged user space task
- Full binary compatibility at API level
- Change hardware aware parts of the kernel
OS Rehosting

- Adapt OS kernel to run as depriviledged user space task
- Full binary compatibility at API level
- Change hardware aware parts of the kernel

L4Linux

- Current mainline version
OS Rehosting

• Adapt OS kernel to run as depriviledged user space task
• Full binary compatibility at API level
• Change hardware aware parts of the kernel

L4Linux

• Current mainline version
• Performance penalty between 2 and 20 percent compared to native
OS Rehosting

- Adapt OS kernel to run as deprivileged user space task
- Full binary compatibility at API level
- Change hardware aware parts of the kernel

L4Linux

- Current mainline version
- Performance penalty between 2 and 20 percent compared to native
- Intrusive modifications to Linux kernel required
Virtualization

- Almost no modifications to virtualized OS kernel required
- With nested paging negligible performance impact
  - up to 30 percent otherwise
- Relies on hardware support
- VT, SVM (and Trustzone)
Virtualization

- Almost no modifications to virtualized OS kernel required
- With nested paging negligible performance impact
  - up to 30 percent otherwise
- Relies on hardware support
- VT, SVM (and Trustzone)

Virtual Machine Monitor (VMM)

- Only with nested paging (VTLB underway)
Virtualization

- Almost no modifications to virtualized OS kernel required
- With nested paging negligible performance impact
  - up to 30 percent otherwise
- Relies on hardware support
- VT, SVM (and Trustzone)

Virtual Machine Monitor (VMM)

- Only with nested paging (VTLB underway)
- Slight modifications to guest kernel (custom virtualized devices)
Virtualization

- Almost no modifications to virtualized OS kernel required
- With nested paging negligible performance impact
  - up to 30 percent otherwise
- Relies on hardware support
- VT, SVM (and Trustzone)

Virtual Machine Monitor (VMM)

- Only with nested paging (VTLB underway)
- Slight modifications to guest kernel (custom virtualized devices)
- Staged virtualization for faithfull virtualization
Achitecture

Hardware (x86, ARM)
µ-Kernel
Infrastructure
IO Mgr MemMgrLoader
Virtual
Network Switch
TPM
Driver
L4Linux (driver provider)
Secure
GUI
TPM
Driver
L4Linux (driver provider)
AutoSAR
Core
Key Mgmt
Signatur Creator
AutoSAR App
AutoSAR App

Applications
Android Stack
Genivi Stack
Trustworthy Document Viewer

unprivileged

Infrastructure
Virtual Network Switch
Secure GUI
TPM Driver
L4Linux (driver provider)

IO Mgr Loader MemMgr

privileged

µ-Kernel

Hardware (x86, ARM)
Proof-of-Concept Implementation
Proof-of-Concept Implementation

- Intel Atom Z510 platform @1.1GHz, 512MB RAM
- Two instances of Android, realtime AutoSAR task
- One Android, one L4Linux with Busybox, realtime AutoSAR task
Further Research
Further Research

Scheduling

Integrate real-time, event driven scheduler with time driven schedule
Further Research

<table>
<thead>
<tr>
<th>Scheduling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrate real-time, event driven scheduler with time driven schedule</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Power Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power efficiency, consider power budget</td>
</tr>
</tbody>
</table>
### Further Research

<table>
<thead>
<tr>
<th>Scheduling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrate real-time, event driven scheduler with time driven schedule</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Power Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power efficiency, consider power budget</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MP / Multicore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symetric and asymmetric multicore systems</td>
</tr>
</tbody>
</table>
### Further Research

#### Scheduling
Integrate real-time, event driven scheduler with time driven schedule

#### Power Management
Power efficiency, consider power budget

#### MP / Multicore
Symetric and asymmetric multicore systems

#### Virtualization
Inter VM communication, performance improvements, hybrid tasks
Important Related Work
Important Related Work

- Experimental Security Analysis of a Modern Automobile
  Koscher, Karl Czeskis, Alexei Roesner, Franziska Patel et al.

- L4Cars
  Kevin Elphinstone Gernot Heise Ralf Huuck Stefan M. Petters Sergio Ruocco
Thank you!