Virtualisierung im Automobil

Lösungen zur Steigerung der Safety und Security automobiler Steuergeräte

Dr.-Ing. Jan Pelzl, Dr.-Ing. Marko Wolf, Dr.-Ing. Thomas Wollinger
escrypt GmbH – Embedded Security, www.escrypt.com

Abstrakt

Techniken zur Virtualisierung, das heißt, mehrere Laufzeitumgebungen parallel auf einer gemeinsamen Hardwareplattform auszuführen, aber streng voneinander abschottet. Virtualisierung ist im Desktop- und Serverbereich inzwischen schon eine Standardtechnologie und wird hier anstatt dedizierter Hardware eingesetzt. Durch heute verfügbare, moderne und hocheffiziente Implementierungen werden die Lösungen zur Virtualisierung auch für Embedded-Applikationen, und dabei insbesondere für den besonders sicherheits- und kostenkritischen Automobilbereich äußerst interessant.

In der vorliegenden Arbeit wird zunächst die Technologie der Virtualisierung kurz vorgestellt, um anschließend auf die Vielzahl möglicher Vorteile und Einsatzgebiete im Automobil, aber auch möglicher Implikationen einzugehen. Im zweiten Teil der Arbeit werden dann verschiedene konkrete Virtualisierungslösungen vorgestellt und auf ihre Eignung im automobilen IT-Umfeld bewertet.

1 Einführung

Die meisten Fahrzeuge waren noch bis vor einigen Jahrzehnten (faktisch bis in der 1990er Jahre), in sich geschlossene, elektromechanische Systeme mit einigen wenigen, isolierten und weitgehend unkritischen IT-Systemen. Heute enthalten bereits Fahrzeuge der Mittel- und Kompaktklasse mehrere Dutzend miteinander verbundenen Mikroprozessoren mit bis zu mehreren hundert Megabytes an Software. Zusätzlich haben diese Fahrzeuge bereits verschiedene externe Kommunikationsschnittstellen in die Fahrzeugelektronik integriert.

Eine weitere Steigerung der Steuergeräteanzahl im Fahrzeug, und die damit verbundene weitere Steigerung des Vernetzungs- und Wartungsaufwands, ist inzwischen weder technisch noch ökonomisch vertretbar [CoChAn02]. Infolgedessen versuchen zukünftige Fahrzeug-IT-Architekturen die Funktionalität vieler einzelner Steuergeräte in einigen wenigen leistungsfähigen Steuergeräten zu vereinen [Frisch04]. Die parallele Ausführung mehrerer Steuergeräteanwendungen auf einer gemeinsamen Hardware ermöglicht die deutlich effizientere und flexiblere Nutzung der stets knappen Hardwareressourcen, verringert Aufwand und Kosten für die Herstellung, den Betrieb und die Wartung von (redundanter) automobiler IT-Hardware sowie notwendiger Verkabelung.

Ein weiterer wichtiger Trend in der automobilen IT-Welt betrifft die fortschreitende Integration von Anwendungen und Schnittstellen, welche oft nicht mehr ausschließlich der Kontrolle des OEMs oder Zulieferers unterliegen. Beispiele sind die Integration von externen, teilweise drahtlosen Benutzerschnittstellen im direkten Fahrzeugumfeld wie USB, Bluetooth oder Wireless-LAN bis hin zu weitreichenden Funkschnittstellen mittels integrierter Mobilfunktechnologie (z.B. GSM oder UTMS). Die bis dato vollständig unter Eigenregie eines OEMs oder Zulieferers entwickelten und betriebenen internen Fahrzeug-IT-Systeme müssen zudem zunehmend auch parallel Anwendungen von Drittanbietern oder Anwendungen des Fahrzeugbenutzers ausführen. So kann auf der zentralen Headunit eines Fahrzeugs neben der fahrrelevanten Parkassistentanwendung des OEMs parallel beispielsweise auch die Bezahlanwendung eines Drittanbieters und/oder die Musikanwendung eines Insassen ausgeführt werden.

Sowohl für die Steuergerätemigration als auch für die Integration von Nicht-OEM-Anwendungen und Schnittstellen sind zusätzliche Mechanismen notwendig, um alle auf der gemeinsamen Hardware parallel ausgeführten IT-Anwendungen zuverlässig gegeneinander abzuschotten und abzusichern. Das bedeutet, dass weder eine zufällige Fehlfunktion (*IT safety*) noch eine gezielte Manipulation (*IT security*) einer der Anwendungen oder Schnittstellen die Funktionalität und die Sicherheit aller anderen, parallel auf dem gemeinsamen Steuergerät ausgeführten Anwendungen unmittelbar negativ beeinträchtigt.

Die Rahmen dieser Arbeit vorgestellten verschiedenen Technologien der Virtualisierung ermöglichen es, eine gemeinsame Hardware effizient und flexibel zu nutzen und dabei gleichzeitig eine wirksame gegenseitige Abschottung der parallel ausgeführten IT-Anwendungen sicherzustellen. Dabei wird zunächst die Technologie selbst sowie mögliche Vorteile und mögliche Implikationen vorgestellt. In einem zweiten Teil werden verschiedene Verfahren zur Virtualisierung konkret vorgestellt und zu ihrem Einsatz / Eignung in automobilen IT-Systemen bewertet.

2 Technologie zur Virtualisierung

Das Basiskonzept der Virtualisierung basiert, wie in Abbildung 2.1 dargestellt, auf einer zusätzlichen Abstraktionsschicht, dem *Virtual Machine Monitor* (VMM), welche/welcher sich zwischen der Hardwarebasis und den darauf ausgeführten Betriebssystemen und/oder Anwendungen befindet. Diese Abstraktionsschicht kann in der Praxis sowohl durch eine spezielle Hardwareerweiterung als auch durch eine zusätzliche Softwareschicht realisiert werden. Die wesentliche Aufgabe des VMM ist es dabei, die real nur begrenzt vorhandenen physikalischen Ressourcen für mehrere, parallel ausgeführte Laufzeitumgebungen, den sogenannten *Virtual Machines* (VM), möglichst frei von Konflikten und Inkonsistenzen gemeinsam nutzbar zu machen, sie zu *virtualisieren*. Die Nutzung der so virtualisierten Hardwareressourcen soll dabei für jede der parallel ausgeführten VMs möglichst so transparent sein, dass diese wie ein alleiniger Prozess auf für sie dezidierter Hardware agieren können. Die gegenseitige Abschottung der VMs sowie die Zugriffssteuerung und das Zugriffsmanagement aller VM-Zugriffe auf die vorhandenen Hardwareressourcen werden durch den VMM realisiert. Das heißt der VMM implementiert alle gültigen und notwendigen Sicherheitsrichtlinien zur Zugriffskontrolle über die gegenseitige Kommunikation, die vorhandenen Anwendungen und Daten sowie über alle verfügbaren Hardwareressourcen. Damit ist der VMM in allen Virtualisierungskonzepten die entscheidende Komponente zur Realisierung der Betriebssicherheit (*IT safety*) und Datensicherheit (*IT security*).

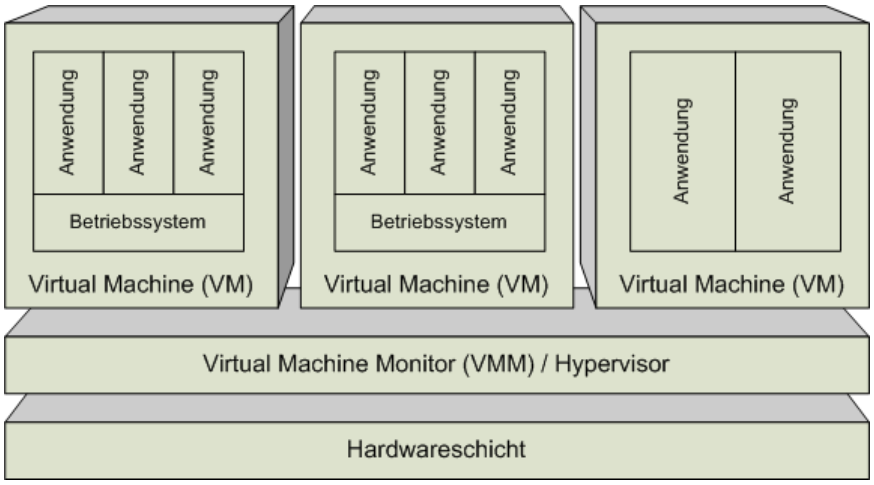

Abbildung 2.1 Prinzipielle Struktur einer virtualisierten IT-Architektur.

2.1 Nutzen und Vorteile

Der Einsatz von Technologien zur Virtualisierung bringt für automobile IT-Architekturen eine Reihe von Vorteilen und Nutzungsgewinnen, von denen eine Auswahl hier nachfolgend kurz vorgestellt werden.

Reduzierung der Hardwarekosten, erhöhte Hardwareeffizienz und höhere Maximalleistung.

Durch die Migration mehrerer Funktionen in eine gemeinsame Hardware kann entweder Hardware ganz eingespart werden oder die vorhandene Hardware zumindest wesentlich effizienter benutzt werden. Die gemeinsame Nutzung von Prozessorzeit, Speicher und Peripherie erhöht die Auslastung, vermindert unnötige Redundanzen, senkt den Verkabelungs- und Wartungsaufwand und ermöglicht viele weitere wertvolle Synergieeffekte unter anderem bezüglich Herstellungs- und Betriebskosten, Administrationsaufwand, Energieverbrauch und Störanfälligkeit. Darüber hinaus steht vor allem kurzzeitig rechenintensiven Anwendungen (z.B. Routenberechnung, kryptographische Operationen etc.) eine deutlich höhere Maximalleistung (*peak performance*) zur Verfügung, als es bei jeweils dezidierten Steuergeräten der Fall wäre.

Vereinfachte Entwicklung und Beherrschbarkeit der Hardwarevielfalt.

Die, durch die Virtualisierung eingefügte, zusätzliche Abstraktionsschicht zwischen Hardware und Steuergeräteanwendung ermöglicht die weitgehende Entkopplung von Steuergeräteanwendung und ausführender Steuergerätehardware. Dies ermöglicht eine von der Hardware weitgehend unabhängige Anwendungsentwicklung auf Basis einer einheitlichen, standardisierten Laufzeitumgebung. Ähnlich dem AUTOSAR-Ansatz [ASAR], kann somit das die Anwendung tatsächlich ausführende Steuergerät sehr flexibel erst im Integrationsprozess (*deployment*) bestimmt werden. Analog zur Anwendungsentwicklung können so auch die Aufwände, und die damit verbundenen Kosten, für das Testen, die Installation und die Wartung deutlich reduziert werden.

Erhöhte IT-Safety.

Die durch eine Virtualisierungslösung mögliche, besonders strenge Abschottung von parallel ausgeführten Laufzeitumgebungen ermöglicht auch eine deutliche Steigerung der IT-Safety gegenüber den vergleichsweise schwächeren Abschottungsmechanismen herkömmlicher (Embedded-) Betriebssysteme [EHHPR05]. Konnte bisher oftmals schon ein fehlerhaftes Programm oder ein fehlerhafter Treiber ausreichen, um die gesamte zugrunde liegende IT-Plattform in Mitleidenschaft zu ziehen, werden voneinander streng isolierte Laufzeitumgebungen nicht mehr direkt von der Fehlfunktion oder dem Absturz einer einzelnen Anwendung oder Treibers beeinträchtigt.

Darüber hinaus können bei einer Virtualisierungslösung über den VMM für eine einzelne VM die Nutzung bestimmter Ressourcen (z.B. Rechenzeit, Speicherverbrauch) fest limitiert werden (Stichwort: Quality-of-Service), um so die Echtzeitfähigkeit anderer Anwendungen und Laufzeitumgebungen der Plattform nicht zu gefährden beziehungsweise ihnen den Umfang der Verfügbarkeit einzelner Ressourcen garantieren zu können.

Steuergerätehersteller können ihre Software zusammen mit ihrer jeweils individuell konfigurierten Laufzeitumgebung als eine VM ausliefern und so Fehler durch eine (z.B. durch den OEM) falsch konfigurierte Laufzeitumgebung oder schwer vorhersehbare Implikationen durch Softwareprozesse anderer Hersteller zu minimieren.

Erweiterte Virtualisierungskonzepte erlauben darüberhinaus das Verschieben ganzer Laufzeitumgebungen (*virtual domains*) über Hardwaregrenzen hinweg, sollte die aktuell ausführende Hardwareplattform plötzlich gestört sein oder zeitweise nur unzureichende Ressourcen liefern. Im Pause-Modus befindliche Software-Funktionseinheiten können, ohne dabei nennenswerte Ressourcen zu beanspruchen, vorgehalten werden, um im Störungsfall für möglicherweise ausgefallene Funktionalitäten direkt zu übernehmen (*hot stand-by*).

Erhöhte IT-Security.

Die für eine Virtualisierung notwendige, besonders strenge Abschottung parallel ausgeführter Laufzeitumgebungen (siehe Abschnitt „Erhöhte IT-Safety.“) kann auch zur Steigerung der IT-Security gegen verschiedenste Softwareangriffe[1] verwendet werden. So können beispielsweise besonders schützenswerte Daten (z.B. kryptografische Schlüssel) und Funktionen (z.B. Tastaturtreiber zur PIN-Eingabe) von denen sie anwendenden Laufzeitumgebungen so isoliert werden, dass (gewollte oder ungewollte) Fehlfunktionen diese nicht gefährden können. So kann beispielsweise das normale Betriebssystem durch eine parallel ausgeführte, isolierte Verschlüsslungsfunktion Daten ver- und entschlüsseln lassen, ohne selbst Zugriff auf die entsprechenden kryptografischen Schlüssel zu haben, so dass auch möglicherweise vorhandene Viren und Trojaner innerhalb des normalen Betriebssystems die kryptografischen Schlüssel nicht ausspähen können. In Kombination mit einer feingranularen, effizienten Zugriffskontrollsteuerung sind nahezu beliebig vielfältige, komplexe IT-Sicherheitslösungen realisierbar [GarWar07].

Multilevel-Security und Multilevel-Safety pro Steuergerätehardware.

Durch den Einsatz von Virtualisierungstechnologie können (und müssen) Prozesse und Anwendungen mit unterschiedlichen Anforderungslevel bezüglich IT-Safety und/oder IT-Security parallel auf einer Hardware ausgeführt werden ohne, dass diese sich gegenseitig beeinträchtigen können.

Mit Hilfe einer VMM mit geeigneten Zugriffsteuerungsmechanismen können – im Gegensatz zu den vergleichsweise schwachen Mechanismen zur Prozessisolation eines Standardbetriebssystems [EHHPR05] – hochkritische Bereiche (z.B. fahrrelevante Anwendungen), kritische Bereiche (z.B. Komfortanwendungen und Fahrhilfen) sowie schützenswerte (z.B. Multimedia und Infotainment) und völlig offene Bereiche (z.B. Internet und eigene Benutzeranwendungen) besonders streng voneinander isoliert (*strong isolation*) und kontrolliert werden, obwohl alle Anwendungen auf einer gemeinsam genutzten Hardware ausgeführt werden. Dies ist ein erheblicher Vorteil gegenüber bisherigen Standardbetriebssystemen, wo das Sicherheitslevel bezüglich IT-Safety und IT-Security von allen Prozessen und Anwendungen eines Steuergerätes vom Sicherheitslevel des am niedrigsten klassifizierten Prozess bestimmt wird, da dieser auch eventuell höher klassifizierte Prozesse und Anwendungen durch potenzielle Sicherheitslücken in Mitleidenschaft ziehen könnte.

Innerhalb einer jeden VM wiederum können dann jeweils beliebig restriktive Zugriffsteuerungsmechanismen verwendetet werden. So müssen keine hoch komplexen und aufwändigen Zugriffssteuerungen automatisch für alle Anwendungen eines Steuergeräts eingesetzt werden. Wenn auf dem gemeinsamen Steuergeräte eine kritische Anwendung ausgeführt wird, können durch den Einsatz von Virtualisierungslösungen nun die jeweiligen internen Zugriffssteuerungen individuell angepasst und für Anforderungen jeder VM ausgeführt werden. Die übergreifende Zugriffssteuerung zwischen den verschiedenen VMs durch den VMM kann hoch effizient realisiert werden.

Erhöhte Flexibilität, Interoperabilität und Rückwärtskompatibilität.

Die Möglichkeit mehrere Gast-Betriebssysteme und somit mehrere Anwendungen für unterschiedliche Betriebssysteme gleichzeitig und nebeneinander auf einer Hardware ausführen zu können, steigert die Flexibilität, Interoperabilität und Rückwärtskompatibilität der verwendeten IT-Plattform erheblich. Bestehende Anwendungen können leicht (zum Teil sogar zur Laufzeit) migriert werden um den Funktionsumfang der Plattform optimal an die jeweilige Situation anzupassen. Ältere, bestehende Anwendungen können in ihrer gewohnten (alten) Laufzeitumgebung, parallel zu neuen Anwendungen in Laufzeitumgebungen ausgeführt werden, ohne aufwändig portiert werden zu müssen (sofern dies überhaupt noch möglich ist). Herkömmliche PC-Programme (Officesoftware, E-

[1] Virtualisierungstechnologien können in der Regel nur begrenzt (z.B. durch physikalische Trennung) gegen Hardwareangriffe schützen.

Mail-Clients, Mediaplayer etc.) können ohne Änderungen parallel neben klassischen Automobilanwendungen im Fahrzeug ausgeführt werden.

2.2 Mögliche Nachteile

Der Einsatz von Technologien zur Virtualisierung in automobilen IT-Architekturen birgt natürlich auch einige mögliche Implikationen, von denen die wichtigsten hier nachfolgend ebenfalls kurz vorgestellt werden.

Potenzielle Leistungseinbußen.

In Abhängigkeit der eingesetzten Virtualisierungslösung (siehe Kapitel 3) ist durch die mögliche Umleitung und Überprüfung (*traps*) von Systemaufrufen, Interrupts und I/O-Zugriffen ein Performanceoverhead im Vergleich zu einer nativen Ausführung ohne VMM möglich. Dieser Performanceoverhead kann – je nach Virtualisierungslösung und zugrunde liegender Basishardware – unterschiedlich hoch ausfallen. Bei hardwaregestützten Virtualisierungsmethoden kann dieser oftmals bis unter die Nachweisgrenze reduziert werden.

Nicht alle Steuergeräte virtualisierbar.

Steuergeräteanwendungen lassen sich umso leichter und wirtschaftlicher auf eine gemeinsame Steuergerätehardware migrieren, je ähnlicher ihre, zuvor dezidiert genutzte, Hardware und Peripherie gewesen ist. Hoch spezialisierte Steuergeräteanwendungen mit Spezialhardware werden sich erheblich schwieriger effizient virtualisieren lassen, genau so wie Steuergeräte mit erhöhtem Störpotenzial, welche beispielsweise Komponenten verwenden, die starke elektromagnetische Felder verursachen.

Ebenso ist die Position und Entfernung von zu virtualisierenden Steuergeräten nicht unerheblich, da die durch die Virtualisierung gewonnenen Hardwareeinsparungen, beispielsweise schnell durch eine eventuell erforderliche längere Verkabelung der notwendigen Sensoren oder Aktuatoren aufgebraucht werden kann.

Letztlich ist auch ein gewisses Minimum an Hardware-Leistungsfähigkeit notwendig, um eine Virtualiserungsmethode überhaupt effektiv einsetzen zu können. So ist der Einsatz von Virtualisierungsmethoden auf 8-Bit oder 16-Bit Architekturen zwar prinzipiell möglich, jedoch heute noch vergleichsweise ineffizient.

Geringere physikalische Redundanz.

Im Vergleich zu mehrfachen, dezidierten Steuergeräten wird die physikalische Redundanz (per Definition) durch Virtualisierung gemindert, so dass eine einzelne Hardwarestörung prinzipiell vergleichsweise mehr Funktionalität beeinträchtigen kann. Nichtsdestotrotz gibt es bereits eine Reihe von Lösungen, wie zum Beispiel die Livemigration (siehe Abschnitt „Erhöhte IT-Safety**.**“), welche diesem Umstand wirkungsvoll entgegenwirken können.

Zugriffskontrollsteuerung notwendig.

Beim Einsatz einer Virtualisierungslösung mit mehreren VMs ist zusätzliche Funktionalität (in Hardware und/oder Software) notwendig, welche die parallelen Zugriffe der einzelnen VMs auf die real nur begrenzt verfügbare Hardware koordiniert und eventuell bestehende Security- und Safety-Richtlinien praktisch umsetzt.

Noch keine zertifizierte Virtualisierungslösung verfügbar.

Die bisher[2] verfügbaren Virtualisierungslösungen sind bislang weder für ihre Sicherheitseigenschaften noch für Echtzeitfähigkeiten validiert oder zertifiziert worden. Eine erfolgreiche Validierung und Zertifizierung der verwendeten Virtualisierungslösung ist für den Einsatz in sicherheitskritischen Fahrzeugbereichen jedoch unabdingbar. Nichtsdestotrotz existiert zumindest für den mikrokernbasierten Ansatz bereits eine erste IT-Security-Zertifizierung nach den Common-Criteria EAL6+ [Higgins08] zur korrekten Umsetzung der Isolation von Laufzeitumgebungen auf Kernelebene (*separation kernel protection profile*) sowie ein laufendes Projekt zur vollständigen formalen Verifikation eines Mikrokernels [L4VER].

2.3 Anwendungen und Einsatzbereiche

Zu den zuvor genannten allgemeinen Vorteilen und Einsatzbereichen, wird im folgenden Abschnitt eine kleine Auswahl möglicher konkreter Einsatzbereiche und Einsatzzwecke für Virtualisierungslösungen im Automobil kurz vorgestellt.

Einsatzorte im Automobil.

Steuergeräte, welche schon heute verschiedene Aufgaben parallel ausführen, können besonders von den zusätzlichen Möglichkeiten und besonderen Eigenschaften von Virtualisierungslösungen profitieren. Das sind demnach insbesondere die Headunit (HU), das Front-Electronic-Module (FEM), die Rear-Seat-Entertainment-Einheit (RSE) und das zentrale Gateway (ZGW).

Einsatzzwecke im Automobil.

Die vergleichsweise strenge Abschottung von parallel ausgeführten Anwendungen und die Möglichkeit Multi-Level-Safety und/oder Multi-Level-Security auf einem Steuergerät zu realisieren prädestiniert den Einsatz von Virtualisierungslösungen vor allem zur strikten Trennung von kritischen Fahranwendungen (z.B. Rückfahrkamera, Einparkautomat) von weniger kritischen und damit meist offeneren und flexibleren Infotainment-Anwendungen (z.B. Mediacenter, Internet, Mobile-Office, Benutzerinstallationen).

Darüber hinaus bieten sich Virtualisierungslösungen vor allem an besonders kritische Security-Funktionalitäten (z.B. Verschlüsselungs- und Verifikationsmodule) und Security-Anwendungen (z.B. PIN-Eingabe, mCommerce) von den übrigen Anwendungen zu isolieren bzw. diese Security-Funktionalitäten aus bestehenden Anwendungen herauszulösen. Die dort involvierten hochsensiblen Informationen (z.B. Schlüssel, PINs oder Zertifikate) können so zusätzlich geschützt und isoliert verwendetet werden, so dass diese zum Beispiel durch mögliche Sicherheitslücken der Anwendung oder ihrer jeweiligen Ausführungsumgebung nicht mehr kompromittiert werden können, da die Anwendung selbst zu keinem Zeitpunkt (d.h. während Erstellung, Verteilung und Ausführung) mehr mögliche sensible Informationen enthält.

Um die – insbesondere möglichen negativen – Eingriffsmöglichkeiten von außerhalb auf das Fahrzeug zu begrenzen, können auch Anwendungen der C2X-Kommunikation, zur Fahrzeugdiagnose (z.B. OBD, Telediagnose) oder zur Integration von Kundenendgeräten (z.B. Mobiltelefone, MP3-Spieler) mittels Virtualisierungslösungen zuverlässig von internen Anwendungen isoliert werden.

Die Möglichkeit mittels Virtualisierung von kritischen Hardwareressourcen auch komplexe Quality-of-Service (QoS) Anforderungen und Verfügbarkeitsgarantien zuverlässig zu gewährleisten, prädestiniert die Verwendung von Virtualisierungslösungen zur strikten Aufteilung und Kontrolle von Kommunikationsressourcen beispielsweise innerhalb des ZGW oder zur Kontrolle von Speicher- und Prozessorressourcen in der HU oder im FSM.

---

[2] Stand: November 2008

Die Erhöhung der Hardwareeffizienz durch Virtualisierungslösungen kann insbesondere im Hochsicherheitsbereich von enormer Bedeutung sein. Statt beispielsweise vier redundante, sich gegenseitig prüfende Hardwareeinheiten zu implementieren, kann die physikalische und funktionale Sicherheit beispielsweise auch durch nur zwei redundante Hardwareeinheiten realisiert werden, welche wiederum jeweils zwei individuelle Laufzeitumgebungen streng voneinander abgeschottet parallel ausführen.

Als abschließenden Einsatzzweck für Virtualisierungslösungen sei noch die schon zuvor erwähnten neuen Möglichkeiten zur Livemigration von Anwendungen und Funktionen von möglicherweise gestörten oder überlasteten Steuergeräten zu „Reservesteuergeräten“ oder Steuergeräten mit noch ausreichenden Ressourcen zu nennen, um auf diese Art eine zusätzliche Redundanz besonders kritischer Fahrzeugfunktionen wie ESP oder ABS zu ermöglichen.

3 Virtualisierungsmethoden

Im folgenden Abschnitt werden die drei wichtigsten Technologien kurz vorgestellt und bezüglich ihrer Eignung in einem automobilen IT-Umfeld bewertet.

3.1 Physikalische Virtualisierung

Bei einer physikalischen Virtualisierungslösung basierend auf physikalischer Trennung wird die Isolation der Laufzeitumgebungen und damit die Funktionalität des VMM (siehe Abbildung 2.1) durch eine Vielzahl von Hardware-Mechanismen erreicht. Dazu können bestimmte Hardwaremodule beispielsweise redundant ausgeführt werden (z.B. Multicore-Prozessoren) oder es werden, wie in Abbildung 3.1 dargestellt, zusätzliche Hardwaremechanismen wie die Erweiterung der Speicherverwaltungseinheit (MPU) eingesetzt, um so mindestens zwei voneinander isolierte Laufzeitumgebungen zu realisieren. Der populärste Vertreter der Virtualisierung auf Hardwareebene aus dem Embedded-Bereich ist die ARM TrustZone-Technologie, welche Prozessor, Arbeitsspeicher und angeschlossene Peripherie in zwei voneinander getrennte Bereiche (*secure world, normal world*) aufteilen kann, welche nur über einen geschützten Dienst (*secure monitor)* und einer neuen ARM-Prozessoranweisung (*secure monitor call*) kontrolliert miteinander kommunizieren können.

Sofern die Hardwaremechanismen des VMM korrekt implementiert sind, können die Isolationsmechanismen des VMM praktisch durch keinerlei Softwarefunktionalität umgangen werden und bieten somit ein besonders hohes und verlässliches Isolationslevel. Darüber hinaus sind Isolationsmechanismen die in Hardware ausgeführt sind in der Regel auch besonders leistungsfähig, so dass kaum mit zusätzlichen Leistungseinbußen durch den Einsatz eines VMM gerechnet werden muss.

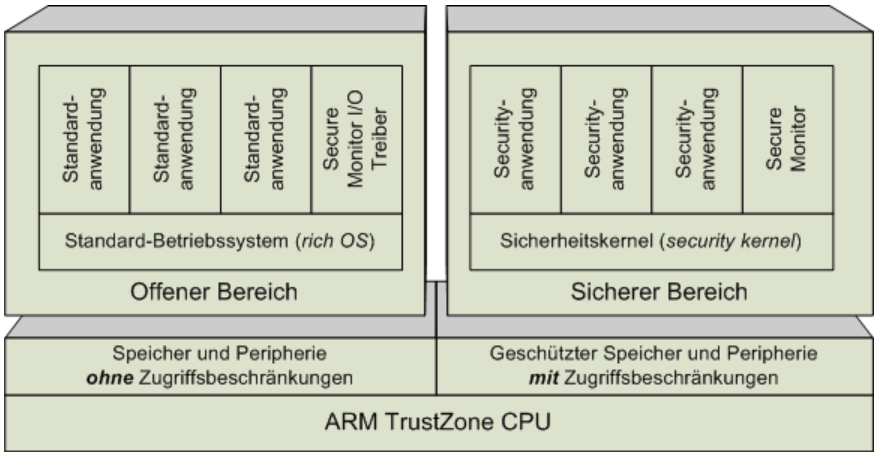

Abbildung 3.1 Physikalische Trennung in einen offenen und einen sicheren Bereich durch die ARM TrustZone CPU-Erweiterung.

Allerdings sind hardwarebasierte VMMs oftmals vergleichsweise teuer und per Definition vergleichsweise inflexibel. Es können auch nur Ressourcen isoliert beziehungsweise virtualisiert werden für die entsprechende Hardwaremechanismen existieren. Die Granularität einer rein hardwarebasierten Virtualisierungslösung ist daher in der Regel begrenzt. Mögliche Einsatzbereiche im Automobil für auf physikalischer Trennung basierende Virtualisierungslösungen sind folglich vor allem Hochsicherheits- und Hochleistungsbereiche (z.B. zentrales Gateway).

3.2 Hypervisorbasierte Virtualisierung

Die, wie in Abbildung 2.1 dargestellt, auf einen Hypervisor basierende Virtualisierungslösungen repräsentieren die „klassische Form" der Virtualisierung. Dabei wird der VMM durch ein minimales Wirtsbetriebssystem realisiert, welches exklusiv im Kernelmode (*privileged mode*) des Prozessors ausgeführt wird. Hierbei behandelt und steuert die Hypervisor-Software alle kritischen Zugriffe auf die virtualisierten Ressourcen. Der Hypervisor realisiert somit die gegenseitige Abschottung der VMs und implementiert die notwendigen Zugriffssteuerungsmechanismen. Populäre Beispiele für Kernelmode-Virtualisierungslösungen im Desktop- und Serverbereich sind beispielsweise Xen [Barham03], KVM [Kivity07] oder Mikrokernel [EHHPR05, Liedtke95]. Im Embedded-Bereich werden kommerziell erfolgreich vor allem die auf Mikrokernen basierenden Virtualisierungslösungen OKL4 [OKL] und Integrity [GHS] eingesetzt.

Indem man versucht den Codeumfang des im Kernelmode ausgeführten Hypervisors so gering wie möglich zu halten[3], kann die auch die Anzahl der möglichen Implementierungsfehler zur Umgehung der Implementierten Isolation und Zugriffssteuerung auf ein Minimum reduziert werden. Besonders kleine Hypervisor (z.B. Mikrokernel) können im Codeumfang teilweise soweit reduziert werden, dass sogar eine formale Verifikation der Korrektheit möglich ist [Shapiro04]. Trotzdem ist das Isolationslevel im Vergleich zu rein hardwarebasierten Virtualisierungslösungen (siehe Kapitel 3.1) etwas geringer. Dafür sind Hypervisor-Virtualisierungen hoch flexibel und vergleichsweise kostengünstig[4], da sie praktisch vollständig in Software realisiert werden können. Die dadurch entstehenden Leistungseinbußen sind heute äußerst gering und bewegen sich praktisch bei 3% bis maximal 10% [YWGK06]. Historisch bedingt existieren heute im Wesentlichen zwei verschiedene Möglichkeiten für eine hypervisorbasierte Virtualisierung. Der etwas ältere Ansatz der Paravirtualisierung und die durch aktuelle Prozessorerweiterungen mögliche hardwaregestützte Virtualisierung werden nachfolgend kurz vorgestellt.

3.2.1 Paravirtualisierung

Paravirtualisierung meint hypervisorbasierte Virtualisierungslösungen ohne das spezielle Virtualisierungserweiterungen im Prozessor (wie z.B. Intel-VT oder AMD-V) notwendig sind. Um mögliche Konflikte durch parallel ausgeführte (nicht privilegierte und daher nicht automatisch abgefangene) Systemaufrufe der verschiedenen VMs zu vermeiden, muss die dort ausgeführte Software so angepasst werden, dass alle kritischen Instruktionen [RobIrv00] statt direkt auf dem Prozessor ausgeführt zu werden, zunächst auf den Hypervisor umgeleitet werden. Das bedeutet Quellcodeanpassungen für alle Softwarekomponenten, welche innerhalb einer VM diese kritischen Instruktionen verwenden. Die Anpassungen der Software (in der Regel sind das einige Basiskomponenten des Betriebssystems) sind zwar relativ gering, benötigen aber (i) Zugriff auf den Quellcode der Software und müssen (ii) jeweils für jede neue Version einer bestehenden Software durchgeführt wer-

---

[3] In der Regel entspricht der Codeumfang eines Hypervisors weniger als 10% eines herkömmlichen Linux- oder Windows-Betriebssystemkernels.

[4] Funktionalitäten in Hardware statt in Software zu realisieren lohnt sich für Halbleiterhersteller aufgrund der hohen Einstiegskosten oftmals erst dann, wenn damit Stückzahlen realisiert werden können, die die allein durch die Automobilhersteller realisierbaren Stückzahlen in der Regel deutlich überschreiten.

den. Das Abfangen und separate Behandeln aller kritischen Instruktionen durch den Hypervisor, verursacht zudem eine nicht unwesentliche Leistungseinbuße.

3.2.2 Hardwaregestützte Virtualisierung

In Anbetracht der bekannten Einschränkungen der Paravirtualisierung verfügen moderne Prozessoren bereits über spezielle Virtualisierungserweiterungen (z.B. Intel-VT oder AMD-V), welche Teile der VMM in Hardware implementieren und somit die möglichen Konflikte durch die Ausführung kritischer Prozessorinstruktionen innerhalb der Prozessorhardware behandeln. Da nun auch kritische Prozessorinstruktionen ohne das Umleiten und Eingreifen des Hypervisors ausgeführt werden können, kann bestehende Software ohne weitere Anpassungen innerhalb einer VM ausgeführt werden. Darüberhinaus kann die Virtualisierung wesentlich effizienter ausgeführt werden, da nun nicht mehr alle kritischen Instruktionen zwangsweise durch den Hypervisor abgefangen und behandelt werden müssen.

3.3 Usermode-Virtualisierung

Als Usermode-Virtualisierung (oftmals auch als Softwarevirtualisierung oder Anwendungsvirtualisierung) werden alle VMMs bezeichnet, welche als reine Softwareanwendung im Benutzermodus (*user mode*) auf einem normalen, vollwertigen Betriebssystem ausgeführt werden. Während bei einer Emulation beziehungsweise Simulation alle Hardwareinstruktionen durch Software behandelt werden (und somit auch auf der Wirtshardware nativ nicht vorhandene Instruktionen simuliert bzw. emuliert werden können), können einige Usermode-VMMs einen Teil der Instruktionen auch direkt auf der vorhandenen Hardware ausführen. Die ausgeführte Gastsoftware muss dabei in der Regel nicht angepasst werden. Aufgrund der Umleitung fast aller Hardwareinstruktionen durch das (nicht minimierte) Wirtbetriebssystem, ergeben sich jedoch im Vergleich zur hypervisorbasierenden Virtualisierung einige Leistungseinbußen. Der entscheidende Nachteil einer Usermode-Virtualisierungslösung ist jedoch der vergleichsweise geringe Gewinn an IT-Sicherheit und Betriebssicherheit/Zuverlässigkeit. Sowohl die IT-Sicherheit (*security*) als auch die Betriebssicherheit (*safety*) aller über die Usermode-VMM ausgeführten Software basiert auf der Sicherheit der darunterliegender Betriebssystemschicht, welche in der Regel durch ein herkömmliches Betriebssystem (z.B. Windows oder Linux) realisiert wird. Die Usermode-VMM ist damit genauso anfällig für alle bekannten Sicherheitslücken und Zuverlässigkeitsstörungen wie das zugrundeliegende Betriebssystem, so dass eine Sicherheitslücke dort oder in einer der parallel ausgeführten Anwendungen gleichzeitig die Sicherheit und Zuverlässigkeit der Usermode-VMM und somit aller darin ausgeführten VMs beeinträchtigt. So erlaubt eine Usermode-VMM im Wesentlichen nur eine Abschottung ihrer ausgeführten virtuellen Maschinen und deren Aktionen gegenüber der Wirtsbetriebssystemumgebung, auch bekannt als *Sandboxing*.

Populäre Beispiele für Usermode-Virtualisierungslösungen sind beispielsweise VMware Workstation [VMW], Microsoft Virtual PC [Kivity07] und Emulatoren wie QEMU oder auch die Java Virtual Machine [LinYel99], welche auch im Embedded-Bereich umfangreich eingesetzt wird.

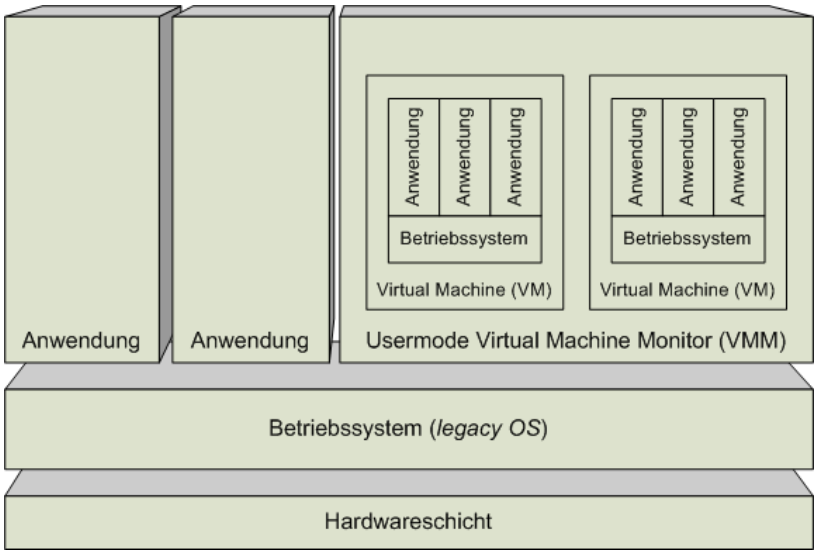

Abbildung 2 Virtual Machine Monitor (VMM) im Benutzermodus.

4 Zusammenfassung und Ausblick

In der Tabelle 5.1 werden die Voraussetzungen, Vor- und Nachteile sowie die möglichen Einsatzgebiete verschiedener Virtualisierungslösungen noch einmal kurz zusammengefasst.

	Voraussetzungen	**Vorteile**	**Nachteile**	**Einsatzgebiete**
Physikalische Virtualisierung	Redundante Hardware oder zusätzliche Hardwarefunktionalität	Maximale Geschwindigkeit und höchstes Isolationslevel, i.d.R keine Softwareanpassungen notwendig	Vergleichsweise teuer, geringe Granularität und Flexibilität, nur redundant vorhandene Hardware virtualisierbar	Hochsicherheitsbereich, Hochleistungsbereich (z.B. zentrales Gateway)
Hypervisor mit Paravirtualisierung	Softwareanpassungen der zu virtualisierenden Laufzeitumgebungen notwendig	Effiziente, flexible Virtualisierung ohne spezielle Virtualisierungshardware möglich	Softwareanpassungen aufwändig oder nicht möglich (*closed source*), nur vorhandene Hardware virtualisierbar	Mittelklasse-Steuergeräte ohne Virtualisierungshardware (z.B. FEM oder REM)
Hypervisor mit hardwaregestützter Virtualisierung	Prozessor mit kompatibler Virtualisierungserweiterung	Effiziente, flexible Virtualisierung ohne Softwareanpassungen	Prozessorerweiterung notwendig, nur vorhandene Hardware virtualisierbar	Hochleistungs-Steuergeräte mit Virtualisierungshardware (z.B. zentrale Multimediaeinheit)

Usermode-Virtualisierung	Vollwertiges Wirtsbetriebssystem notwendig	Beliebig Hardwarekonfigurationen einschließlich verschiedener CPUs simulierbar	Vergleichsweise ineffizient, Sicherheit der VM basiert auf der Sicherheit des Wirtsbetriebssystems	Abschottung nicht vertrauenswürdigen Anwendungen innerhalb einer vertrauenswürdigen Laufzeitumgebung (z.B. Internetzugang)

Tabelle 5.1 Überblick über Voraussetzungen, Vor- und Nachteile sowie möglicher Einsatzgebiete verschiedener Virtualisierungslösungen im Automobil.

Obwohl, soweit den Autoren derzeit bekannt, bis auf wenige Ansätze zum *Sandboxing* von einzelnen kritischen Anwendungen (z.B. Portalzugänge via GSM) heute noch keine der hier vorgestellten Virtualisierungslösungen in der automobilen Praxis eingesetzt werden, sind Virtualisierungslösungen schon jetzt ein Standard-Technologiebaustein für die meisten Fahrzeug-IT-Architekturen der Generation 2010ff. Der zunächst hardwareunabhängige, rein modul- und funktionsorientierte AUTOSAR-Ansatz („Automotive Open Software Architecture“) für eine offene, flexible und standardisierte automobile Softwarearchitektur [ASAR] wird beispielsweise kaum ohne einen zuverlässigen und effizienten Isolationsmechanismus praktisch realisierbar sein.

Allein die fortschreitende Integration von IT-Anwendungen im Automobil, welche schon heute nicht mehr vollständig unter der Kontrolle des jeweiligen OEMs/Zulieferers liegen (z.B. Nutzergeräteanbindung, Benutzersoftware oder das Internet) wird schon in naher Zukunft nicht mehr nur mittels isolierter, dedizierter Hardware realisiert werden können. Genauso wie die stetig fortschreitende Migration von einzelnen Steuergeräten in zentrale leistungsstarke IT-Einheiten , welche ohne den Einsatz der hier vorgestellten Virtualisierungslösungen kaum effizient, zuverlässig und sicher umzusetzen sein wird. Hierbei versprechen insbesondere die Kombination aus Hardware- und Softwaremechanismen zur Virtualisierung maximale Flexibilität und Effizienz. So kooperiert etwa der Prozessorhersteller Intel bereits mit dem Linux-Entwickler WindRiver mit dem Ziel die Virtualisierungserweiterung von Intels Automotive-Prozessors „Atom“ zusammen mit der Hypervisor-Virtualisierungslösung des Automotive-Linux von WindRiver zu verknüpfen.

Insofern ist der Einsatz von Virtualisierungslösungen im Automobil in unmittelbarer Zukunft sowohl eine unverzichtbare Notwendigkeit, als auch eine große Chance zur Steigerung der Safety und Security automobiler IT-Anwendungen.

Referenzen

[ASAR] The Automotive Open System Architecture (AUTOSAR). www.autosar.org.

[Barham03] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield. Xen and the Art of Virtualization. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles*. New York, NY, USA, 2003.

[CoChAn02] E. Coelingh, P. Chaumette, M. Andersson. Open-Interface Definitions for Automotive Systems – Application to a Brake-By-Wire System, Technical Paper 2002-01-0267, SAE International, March 2002.

[EHHPR05] K. Elphinstone, G. Heiser, R. Huuck, S. M. Petters, S. Ruocco. L4Cars. In *3rd Workshop on Embedded Security in Cars (escar)*. November, 2005.

[Frisch04] H.-G. Frischkorn. Automotive Software – The Silent Revolution. In *Workshop on Future Generation Software Architectures in the Automotive Domain*. San Diego, USA, 2004.

[GarWar07] T. Garfinkel, A. Warfield. What Virtualization can do for Security. In *The USENIX Magazine, Volume 32, Number 6*. December 2007.

[GHS] Green Hills Software (GHS). The Integrity Embedded Operating System. www.ghs.com.

[Higgins08] Kelly Jackson Higgins. In www.darkreading.com/security/app-security/showArticle.jhtml?articleID=212100421, November 2008.

[Kivity07] A. Kivity, Y. Kamay, D. Laor, U. Lublin, A. Liguori. KVM: The Linux Virtual Machine Monitor, In *Proceedings of Linux Symposium*. Ottawa, Canada, 2007.

[L4VER] L4.verified Project. http://ertos.nicta.com.au/research/l4.verified/

[Liedtke95] J. Liedtke. On Microkernel construction. In *Proceedings of the 15th ACM Symposium on Operating Systems Principles*. Copper Mountain, CO, USA, 1995.

[LinYel99] T. Lindholm, F. Yellin. Java(tm) Virtual Machine Specification. Addison-Wesley Professional, 1999.

[OKL] Open Kernel Labs (OKL). The OKL4 Embedded Operating System. http://wiki.ok-labs.com/.

[RobIrv00] John Scott Robin, Cynthia E. Irvine . Analysis of the Intel Pentium's Ability to Support a Secure Virtual Machine Monitor. In *Proceedings of the 9th USENIX Security Symposium.* Denver, Colorado, USA, August 2000.

[Shapiro04] J. Shapiro, M. S. Doerrie, E. Northup, S. Sridhar, M. Miller. Towards a verified, general-purpose operating system kernel. In *Proceedings of NICTA FM Workshop on OS Verification*. National ICT Australia, 2004.

[YWGK06] L. Youseff, R. Wolski, B. Gorda, C. Krintz. Evaluating the Performance Impact of Xen on MPI and Process Execution For HPC Systems. In *Proceedings of the International Workshop on Virtualization Technologies in Distributed Computing*, Tampa, FL, USA, 2006.

[VMW] VMware Inc.VMware Workstation. http://www.vmware.com/products/ws/

Abbildungen und Tabellen