Virtualisierung in ressourcenbeschränkten mobilen und eingebetteten Systemen

Jochen Streicher[1], Olaf Spinczyk[1], Bernhard van Bonn[2], Bernd Schmidt[2] und Boris Kißner[2]

[1]TU Dortmund
{jochen.streicher,olaf.spinczyk}@tu-dortmund.de

[2]Fraunhofer Institut für Materialfluss und Logistik
{bernhard.van.bonn,bernd.schmidt,boris.kissner}@iml.fraunhofer.de

1. Motivation

Virtualisierung mobiler und eingebetteter Systeme erfährt seit einiger Zeit wachsendes Interesse. Mit der Logistik haben wir eine Domäne untersucht, die entsprechende Hardware einsetzt. Dabei zeigte sich, dass Virtualisierung für diese Domäne aus mehreren Gründen interessant ist. Hardware kann besser ausgenutzt und flexibler für verschiedene Aufgaben eingesetzt werden, was zu höherer Investitionssicherheit und geringeren Ausgaben führt. Die durch Virtualisierung gegebene Isolation ist notwendig, damit auch Softwaresysteme sich potentiell gegenseitig misstrauender Teilhaber an logistischen Prozessen auf derselben physischen Plattform integriert werden können. Wünschenswert ist auch die Möglichkeit der Migration virtueller Maschinen auf oder von Mobilgeräten, beispielsweise wenn die Benutzerumgebung samt Applikationen und persönlicher Daten auf ein anderes Gerät übertragen werden soll. Auch für einzelne Applikationen wäre eine einfache Installation und Deinstallation dieser in Form von VM-Images (*virtual appliances*) sinnvoll, da in einer logistischen Transportkette viele verschiedene Unternehmen beteiligt sind, die jeweils eigene Softwaresysteme einsetzen.

Dank der inzwischen teilweise enormen Leistungsfähigkeit mobiler Systeme ist Virtualisierung auf vielen dieser Systeme prinzipiell einsetzbar. Dennoch unterliegen diese Geräte Einschränkungen, die PC- oder Serversysteme nicht betreffen. Hardwareressourcen wie Rechenleistung, Arbeitsspeicher und nichtflüchtiger Speicher fallen wesentlich knapper aus. Weiterhin ist die Netzwerkanbindung mobiler Geräte, insbesondere über Mobilfunknetze, weniger verlässlich, nicht überall verfügbar und bietet i.A. eine wesentlich geringere Bandbreite. Die Bereitstellung von Software über diese Netze kann kaum über VM-Images mit Größen im Gigabyte-Bereich geschehen.

Mit feiner werdender Granularität, d.h. im Extremfall eine virtuelle Maschine pro Anwendung anstatt für komplette Anwendungsdomänen, werden die Hardware-Anforderungen die Kapazität auch der leistungsfähigeren Geräte übersteigen, da für jede VM ein eigenes Betriebssystem benötigt wird. Je weniger Anwendungen mit einem Betriebssystem zusammen in einer VM laufen, umso größer ist der Anteil des Betriebssystems am Ressourcenbedarf der VM und der Anteil an Betriebssystemfunktionalität, die nicht benötigt wird. Dies gilt insbesondere, wenn gängige mobile, auf Linux basierende Betriebssysteme oder MS-Windows-Derivate, wie sie in der Logistik üblich sind, eingesetzt werden. Energie ist i.d.R ebenfalls eine beschränkte Ressource in diesen Systemen, deren Verbrauch von uns im Rahmen dieses Beitrags jedoch nicht als eigenständiges Problem behandelt, sondern als natürliche Folge der Nutzung anderer Ressourcen betrachtet wird, insbesondere des Prozessors und des mobilen Netzwerks.

2. Existierende Ansätze

Während Virtualisierung also in erster Linie zu gesteigerten Hardware-Anforderungen führt, eröffnen sich gleichzeitig auch Möglichkeiten, diese zu reduzieren. Diese können dabei sowohl von “außen”, d.h. von Seiten der VM-Umgebung bzw. des Hypervisors, als auch von “innen” d.h. von Seiten der Gastbetriebssysteme wahrgenommen werden. Als wesentliche Konzepte zur Verminderung des Ressourcenbedarfs sind auf der einen Seite das *Sharing* zwischen virtuellen Maschinen, auf der anderen Seite die *Maßschneiderung* einzelner VMs zu nennen. Für beide Konzepte gibt es sowohl statisch als auch dynamisch angewandte Ausprägungen.

Sharing bezeichnet die gleichzeitige gemeinsame Verwendung von Teilen des physikalischen Speichers zwischen virtuellen Maschinen bei gleichzeitiger Sicherstellung der Isolation, d.h. diese Teile müssen entweder *read-only*- oder *copy-on-write*-Semantik besitzen. In einigen existierenden Virtualisierungslösungen wird dies bereits gemacht, indem z.B. inhaltlich identische Seiten innerhalb und zwischen VMs geteilt werden. [3] Mit virtuellen Massenspeichern

kann ähnlich verfahren werden, sowohl von außen [4], als auch von innen. Idealerweise muss bei der Übertragung neuer VM-Images nur ein wesentlich kleineres Differenzimage übertragen werden. Sharing funktioniert natürlich nur, wenn sich die verwendeten Gastbetriebssysteme entsprechend ähnlich sind.

Nicht diesen Beschränkungen unterliegt die *Maßschneiderung* von Betriebssystemen auf die Anforderungen der Anwendung, welche darüber hinaus noch der Reduktion der Prozessornutzung dient. Gängige Betriebssysteme für mobile Geräte bieten zwar heute schon einen gewissen Grad an Maßschneiderbarkeit, jedoch enthält das Resultat i.d.R. immer noch unbenötigte Funktionalität, die unnötig Hardware-Ressourcen, insbesondere Prozessorleistung in Anspruch nimmt. Quelloffene Betriebssysteme lassen sich zwar "von Hand" optimieren, was aber sehr zeitaufwändig ist, da es für jede Anwendung erneut gemacht werden muss. Daher werden wiederverwendbare Ansätze benötigt. Hierbei kann man in zwei Richtungen vorgehen: Zum einen kann man versuchen, die Elimination unbenötigter Funktionalität in existierendem Betriebssystemcode zu automatisieren, zum anderen kann man einen konstruktiven Ansatz verfolgen und Betriebssysteme (oder Teile davon) von Grund auf maßschneiderbar entwerfen, wobei beide Richtungen kombiniert werden können.

Aus dem Bereich der eingebetteten Systeme stammen einige Ansätze zur Automatisierung der Reduktion von Größe [1], Taktzyklen- [2] und damit auch Energiebedarf von Betriebssystemcode, die teilweise statisch und teilweise zur Laufzeit angewandt werden. Diese Techniken sind jedoch teilweise weniger gut für Closed-Source-Betriebssysteme geeignet oder nur teilweise automatisierbar.

In der anderen Richtung existieren ebenfalls einige Arbeiten, die sich mit Methoden zum Entwurf von Betriebssystem-Produktlinien beschäftigen, die sich mit wesentlich feinerer Granularität maßschneidern lassen. Die Ableitung eines konkreten maßgeschneiderten Systems erfolgt hier nicht nur durch rein codebasierte Techniken, sondern auch auf Basis abstrakter Merkmale, womit auch architekturelle Eigenschaften maßgeschneidert werden können, z.B. die Isolation innerhalb des Gastbetriebssystems, die u.U. nicht benötigt wird.

3. Offene Fragen

Im Bereich des Server-Hostings kann durch Sharing viel Speicherplatz eingespart werden, da dort oft viele gleichartige virtuelle Maschinen verwendet werden, was in diesem Ausmaß in eingebetteten oder mobilen Systemen wohl nicht der Fall sein wird. Daher stellt sich in dieser Domäne die Frage, ob Sharing überhaupt sinnvoll ist, insbesondere, da es mit feingranularer Maßschneiderung kollidiert. Letzteres führt zu der Frage, ob oder wie man beides kombinieren kann.

Viele dieser Ansätze sind für sich allein insofern beschränkt, als dass Sie entweder auf einer "äußeren Sicht" basieren, d.h. ohne Wissen über die Interna der Gastbetriebssysteme zu besitzen, oder auf einer "inneren Sicht", d.h. ohne sich der Tatsache bewusst zu sein, in einer VM zu laufen. Es ist deshalb weiterhin zu untersuchen, inwieweit sich diese durch einen Informationsaustausch zwischen Gastsystem und Hypervisor, der über Paravirtualisierung hinausgeht, verbessern lassen. Bisher existiert so etwas höchstens als Ad-hoc-Erweiterung. Ein Beispiel hierfür ist der "Balloon-Treiber" [3], ohne den der durch Sharing gewonnene Hauptspeicher gar nicht genutzt werden könnte. Weiterhin stellt sich die Frage, ob existierende Methoden zur dynamischen Maßschneiderung unter Ausnutzung der Virtualisierung verbessert oder vereinfacht werden könnten.

Je besser maßschneiderbar ein Betriebssystem ist, und umso mehr Konfigurations- bzw. Spezialisierungsmöglichkeiten damit existieren, desto aufwändiger werden manuelle arbeiten, die damit verbunden sind. Essentiell für einen verbreiteten Einsatz dieser Techniken sind daher automatische Verfahren die einerseits die Anforderung der Anwendung und andererseits die Spezialisierungsmöglichkeiten im Betriebssystem analysieren.

Grundsätzlich ist zu sagen, dass es verschiedene Ansätze für spezielle Probleme gibt, für die wir untersuchen wollen, inwiefern diese sich a) sinnvoll kombinieren und b) so automatisieren lassen, dass sie in ein Gesamtsystem integriert werden können, das von einem Anwendungsentwickler, der sich keine Gedanken um den Ressourcenverbrauch seines Betriebssystems machen möchte, genutzt werden kann um eine schlanke *virtual appliance* für ressourcenbeschränkte Systeme zu erhalten.

Literatur

[1] Dominique Chanet, Bjorn De Sutter, Bruno De Bus, Ludo Van Put, and Koen De Bosschere. Automated reduction of the memory footprint of the linux kernel. *Trans. on Embedded Computing Sys.*, 6(4):23, 2007.

[2] D. McNamee, J. Walpole, C. Pu, C. Cowan, C. Krasic, A. Goel, P. Wagle, C. Consel, G. Muller, and R. Marlet. Specialization tools and techniques for systematic optimization of system software. *ACM Transactions on Computer Systems (TOCS)*, 19(2):217–251, 2001.

[3] C.A. Waldspurger. Memory resource management in VMware ESX server. In *OSDI'02*, 2002.

[4] A. Warfield, R. Ross, K. Fraser, C. Limpach, and S. Hand. Parallax: managing storage for a million machines. In *HOTOS'05*, pages 4–4.