Virtualization for automotive IVI systems

As the demand for modern in-vehicle infotainment systems grows, automakers are increasingly looking toward virtualization as a solution to bridge the gap between consumer and automotive electronics.

by V. Saminathan and G. Swaminathan

In the past, automotive audio systems were characterized by simple functionality, a simple tuner, minimal user interface, and little to no two-way communication channels. They were also closed, in the sense that all of the software was loaded pre-sale by the manufacturer and normally remained unchanged for the lifetime of the device. The amount of software was small.

Many modern in-vehicle infotainment (IVI) systems, however, are very different. Today’s systems have sophisticated user interfaces, consisting of input keys or a touch screen, rearview cameras, audio, and high-resolution video outputs. Additionally, they combine many functions, including voice and data communication, productivity tools, media players, and games. They also support different wireless communication modes, including Wi-Fi, Bluetooth, and infrared. Finally, they allow users to load data and even application programs or “apps.” The total amount of software running on today’s devices is complex and large, measuring into the millions of lines of code.

Need for virtualization

The gap between consumer electronics and automotive electronics started growing wider because of the rapid consumer adoption of smartphones during the early 2010s. Consumers expect to do everything in their car that they do on their mobile devices—including seamless connectivity to the external world. The automotive industry was conservative for many reasons during the past, but now it must try to bridge the gap between consumers’ expectations for infotainment features and the current performance of today’s automotive electronics.

The key problems faced by automakers with respect to IVI electronics are:

- Firmware updates: IVI feature sets are rapidly outdated when compared to consumer electronics products
- In-vehicle security: IVI becomes the gateway for the in-vehicle network, which triggers the need for a firewall between the in-vehicle network and the outside world.

To address the above problems, “sandboxing” (virtualization) is emerging as a key solution.

The concept of virtualization originated in mainframes and is used extensively in the personal computer domain. It has recently picked up in the embedded electronics world, as well.

There are three different types of virtualization techniques:

- Virtual machines: In a virtual machine (VM), the hypervisors run on top of the host operating-system environment. With the hypervisor layer as a distinct second software level, guest operating systems run at the third level above the hardware. (Examples: VMware products, Oracle VirtualBox, QEMU)
- Para-virtualization or bare-metal virtualization: With para-virtualization, the hypervisors run directly on the host’s hardware to both control the hardware and manage guest operating systems. A guest operating system runs on another level above the hypervisor. (Example: COQOS from OpenSynergy)
- Containers virtualization: A method where the kernel of an operating system allows for multiple isolated user-space instances, instead of just one. Virtualization techniques each have their own advantages and disadvantages, although para-virtualization is the most widely used on automotive embedded platforms today.

Virtualization benefits

The two primary benefits of virtualization are related to firmware updates and in-vehicle security.

Firmware updates: Virtualization has opened new doors when it comes to IVI design techniques. Using virtualization, an IVI architecture can be separated into critical and non-critical partitions. The critical partition takes care of vehicle network communication and other critical software that requires an RTOS (real-time operating system) or thin OS. On the other hand, media, connectivity, voice features, and other UI-related features are
Virtual Machines are one of three types of virtualization techniques.

<table>
<thead>
<tr>
<th>Guest Application</th>
<th>Guest Application</th>
<th>Guest Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guest Operating System</td>
<td>Virtual-Machine Monitor (VMM)</td>
<td>Host Operating System</td>
</tr>
</tbody>
</table>

Para-Virtualization is another widely used technique.

<table>
<thead>
<tr>
<th>Guest Application</th>
<th>Guest Application</th>
<th>Guest Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guest OS &amp; Root File System</td>
<td>Guest OS &amp; Root File System</td>
<td>Host Operating System</td>
</tr>
</tbody>
</table>

In Containers Virtualization, the kernel of an operating system allows for multiple isolated user-space instances.

placed in the non-critical partition. This partition runs on a high-end, consumer-based OS such as Linux or Android. This architecture also makes it possible for the firmware of one partition to be updated without affecting the other partition. This benefits OEMs by enabling them to keep the non-critical (application) partition up-to-date without affecting the critical partition.

In-vehicle security: Virtualization can be used to enhance security. A virtual machine encapsulates a subsystem so that its failure cannot interfere with other subsystems. In an IVI, for example, the in-vehicle CAN (controller area network) communication stack is of critical importance. If the stack were subverted by an attacker, the IVI could interfere with other critical CAN nodes in the vehicle network. In an extreme case, the vehicle network could be compromised, which could result in a vehicle becoming stranded in the middle of the road. Similarly, an encryption subsystem needs to be strongly shielded from potential threats so that sensitive user and vehicle information can be protected from hackers. This is a significant challenge for a system running millions of lines of code, many of which are security-critical. The high-level OS is particularly vulnerable to attacks in open systems (which allow owners to download and run arbitrary programs), and is large enough to contain hundreds or thousands of bugs. In the absence of virtualization, the high-level OS runs in “privileged mode,” which means that, once compromised, it can attack any part of the system. With virtualization, the high-level OS is “de-privileged” and unable to interfere with data belonging to other subsystems. Additionally, its access to the processor can be limited to ensure that critical components are not compromised by bugs or other unscrupulous code.

Limits of virtualization

Virtualization has its limits related to software complexity, integration, security policies, and trusted computing base.

Conclusions

For the ARM Cortex-A9 processor, para-virtualization is generally seen as the best option in the embedded domain. Similarly, for the Cortex-A15 processor, a thin, para-virtualized hypervisor can continue to serve as the foundation for embedded IVI systems, performing resource management (memory, devices, energy, and global scheduling) and facilitating secure communication and resource sharing among guest operating systems.

On the other hand, ARM’s A15 core also comes with hypervisor mode, which enables hardware-assisted virtualization. Focusing on this strategy, para-virtualization suppliers such as OpenSynergy are also extending their hypervisor mode to accommodate ARM Cortex-A15 processor capabilities, including hardware-assisted virtualization, 40-bit addressing, and the latest ARMv7-A instruction set. Hardware-assisted virtualization reduces the maintenance overhead of para-virtualization, as it reduces the changes needed in the guest operating system.

Going forward, these types of virtualization architectures are poised to provide the best options for designing world-class software for tomorrow’s connected vehicles.