I. INTRODUCTION

Future connected car implementations require secure and cost-efficient platform. The V2X connectivity applications require secured access to the radio and an easy-to-use platform to create applications. Car makers are adding a lot of new functionality into the instrument cluster and infotainment units. Security and safety have always been important, but even more now, because systems face new digital threats.

II. REQUIREMENTS

The U.S. Department of Transportation (DoT) [1] and the U.S. National Highway Traffic Safety Administration (NHTSA) [2] outline the requirements for V2X connectivity of the vehicles. Security and privacy are major concerns of the V2X connectivity. To meet these requirements, we suggest that the communication module needs to be fully isolated at hardware-level. In addition to security and privacy concerns, the NHTSA and DoT are concerned about the cost of the devices to the consumers. The European Union is also planning to mandate implementation of V2I applications.

Miller and Valasek[3] exposed many vulnerabilities in CAN bus implementations in recent cars. Similar hardware-level isolation is required also for CAN bus connectivity. Earlier access to CAN bus required physical presence, but increased connectivity of cars raises this threat to a new level. The attacks are inevitable and those should be prevented and contained as efficiently as possible.

Car manufacturers are adding new kind of user interfaces that integrate information across instrument cluster and infotainment unit. The information displayed needs to be correct and critical information needs not to suffer from partial failure or a security breach in the system. Also external parties want to create applications for cars, such as governments, service providers and insurance companies, and that will create new security concerns. Added functionality make systems more complex and more vulnerable.

Fig. 1. Overview of the system.

Many applications in cars require real-time capability from the system. The increased complexity of car computers will increase costs and thus a platform connected cars needs to be very cost-efficient. Adding new hardware devices will increase the cost, combining several devices into one can decrease the cost.

III. ARCHITECTURE

The architecture we propose is based on dual or quad-core ARM-Cortex CPU with TrustZone and proven open source components. The virtualization is used to sandbox the system into partitions and to increase overall efficiency of the system. The Fig. 1 shows the components of the system. The Secure OS is the most protected partition of the system, and it contains CAN and V2X radio access. The Core OS is a real-time operating system that can be used only by car manufacturers and their partners to implement critical applications. The Rich OS is a Linux distribution based on Mer distribution. The Rich OS is used to implement infotainment functionality and 3rd party access to it can be allowed.

The virtualization is done in two layers. The ARM Trust-Zone first divides the system into protected and non-protected partitions. The virtualization layer in non-protected partition is used to further divide the system to the Core OS and the Rich OS. The Fiasco OC is used as a kernel in Secure OS and in the virtualization layer. The Fiasco OC was chosen as it is available for many hardware platforms, actively maintained and contains L4Linux for virtualization. The RTLinux enhanced with L4Linux real-time enhancements is used as the Core OS, that is known to provide good real-time performance [4]. Virtualization allows to implement multiple systems that share resources to improve cost-efficiency.

The security is divided into three fully independent layers. A breach inside one of the layers does not result into a breach.
of inner layers and the virtualization layers will contain the attack. The layer below is also more controlled environment, making it harder to break. The ARM TrustZone is recognized by the DoT[1] as a hardware-level isolation technology. The design of the L4 kernel features minimal Trusted Computing Base (TCB), that allows more rigorous verification of most critical software. The formal verification of the seL4[5] shows that L4 kernel can be formally verified.

The CAN and V2X connectivity is implemented inside the Secure OS. Both CAN and V2X connectivity require full isolation from the rest of the system. TrustZone restricts the access to hardware only from protected partition. The hardware device driver, connectivity protocol stack and CAN, V2X services are implemented in the Secure OS (see Fig. 2). The services provide a secure interface to the connectivity interfaces. Cellular modem implementations in smart phones use similar design.

The Software Development Kits (SDKs) are based on Qt Creator SDK. The SDKs allow creation of software for both Rich OS and Core OS, and thus it is possible to flexibly create hybrid applications that are partially real-time and partially running in the Rich OS environment.

Few automotive SoCs contain two separate graphics accelerators. In that scenario, one accelerator can be dedicated to the Core OS and it can offer real-time guaranteed rendering. The both OSes can render to same display in secure way that allows more flexibility.

IV. APPLICATIONS

Three example applications in Fig. 3 highlight the features of the platform.

Instrument cluster and Head Unit with secure V2X communication: The implementation of the instrument cluster in the Core OS and the head unit in the Rich OS sandboxes the systems and allows more flexibility in the Rich OS. Delivery of the most critical V2X messages can be ensured through the instrument cluster.

Distance based cruise control with multimodal output: The implementation of the cruise control in the Core OS allows it to function independently and in real-time. It accepts input only through CAN bus and provides information to the Rich OS. The Rich OS can use the cruise control data for various applications and visualize it.

3rd party road toll payment application: The road toll payment application requires the frontend to communicate with the payment system and access to the V2I protocol stack to communicate with the infrastructure. The payment protocol is implemented in the Rich OS, thus it can be also provided by a 3rd party. The application communicates with the V2I protocol stack, that is implemented in the Secure OS.

V. CONCLUSION

The components for creating safe and secure open source based V2X implementations exist today. Many companies, such as Bosch, Sygic, Continental, are developing proprietary systems that also utilize virtualization technologies. The open source based solution is more cost-efficient and is built on proven technologies.

Further development include replacing the L4Linux with a hypervisor. The formal verification work based on results of the seL4[5] needs to be applied to Fiasco.OC and further extended to V2X and CAN protocol stacks. The split of applications into layers of the system requires design work and experimentation. It also includes selection of secure and efficient inter process communication (IPC) mechanism.

The platform is viable option even for self driving cars, and it can be applied to other kind of vehicles that require safe and secure platform with developer friendly SDKs.

REFERENCES