2015 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
VEHICLE ELECTRONICS AND ARCHITECTURE (VEA) TECHNICAL SESSION
AUGUST 4-6, 2015 – NOVI, MICHIGAN

MULTIPLE INDEPENDENT LEVELS OF SECURITY (MILS) NETWORK REFERENCE ARCHITECTURE

Mr. David Jedynak
Chief Technology Officer, COTS Solutions
Curtiss-Wright Defense Solutions
Austin, TX

ABSTRACT

This paper presents the MILS Network Reference Architecture, including the added benefit of safety critical domains for a completely integrated mixed security and mixed safety hardware and software reference architecture for platforms, driving to minimal SWaP and maximum flexibility in the use of vehicles. Included are specific examples of techniques, application to specific systems, and performance concerns. Overall SWaP-C metrics are discussed. In addition, the enabled operational capability of user-based role and security level reconfiguration is explained in detail.

INTRODUCTION

Information Security on vehicles is an increasingly complex problem. The budgetary and capability set drive to do more with less and to be more flexible in operations demands that defense platforms serve a number of roles with minimal reconfiguration. A single platform may be intended to serve any number of configured roles including combat, command and control, medical evacuation, personnel transport, cargo transport, or special operations. In addition, these same platforms may be destined for allied, coalition, or in-country use by forces which may or may not be trusted to various levels. Finally, in some cases, these platforms may be compromised and/or captured. Simply declaring a platform and its onboard information systems as System High is no longer a viable option going forward to ensure the greatest operational flexibility and commonality for assets.

Size, Weight, Power and Cost (SWaP-C) constraints on platforms make it impossible to use fixed-installation techniques for managing access and compartmentalization (e.g. SCIF). Separate hardware and networks dedicated to information within multiple security domains would dramatically over-burden a platform's available budgets. Rather than attempting to separate a vehicle in multiple physical partitions, software and network security techniques need to be used to share the physical hardware in service of multiple roles and personnel.

The traditional application of Multi-level Security (MLS) systems which attempt to implement Bell-LaPudula (write-up / read-down) and Biba (compartmentalization) models in security kernels are functionally impossible to formally validate above EAL4, especially as the number of processes and end-uses increase. Until relatively recently, security kernels along with trusted networking labeling were the only real approaches to shared hardware infrastructure. With the advent of processing virtualization and various network management and security techniques, Multiple Independent Levels of Security (MILS) with a Type I Hypervisor (bare metal) serving as a separation kernel provides a viable method by which to both implement and formally validate (up to EAL 7) a mixed security infrastructure. In addition, the use of Commercial Systems for Classified (CSfC) techniques for securing the network can be used.

The ultimate goal of this paper is to demonstrate a viable path forward for mixing security levels, including safety critical systems, to enable platforms to serve multiple roles, functions, and personnel with effective use of SWaP-C, minimized acquisition investment, and minimal operator burden.

MULTI-LEVEL SECURITY USING SECURITY KERNELS

The traditional approach to Multi-Level Security – the mixing of multiple security levels within a single operating context – is to use a *security kernel*, which enforces rules of interactions (e.g. BLP and Biba) between multiple users and data within a single operating environment. The security kernel relies on the use of attributes – or labels – for files, processes, and users to make access decisions. The logical extension of this is to use labels on network traffic as well, as shown in Figure 1.

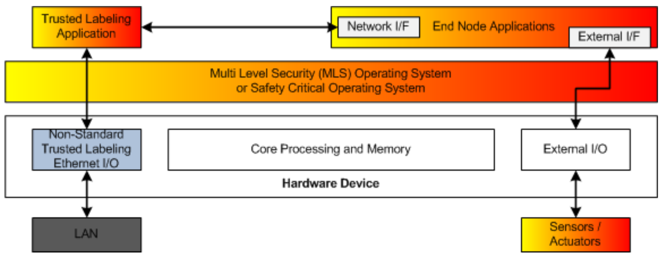
Trusted Labeling Application
Network I/F
End Node Applications
External I/F
Multi Level Security (MLS) Operating System or Safety Critical Operating System
Non-Standard Trusted Labeling Ethernet I/O
Core Processing and Memory
External I/O
Hardware Device
LAN
Sensors / Actuators

Figure 1: Traditional Multi-Level Security Model with Trusted Labeling and Security Kernel

Although this solution can and has been implemented, it is functionally difficult to verify. The very same rule of thumb, *Metcalfe's Law*, which tells us the "value" of a network is the square of the number of nodes – is a double-edged sword. Metcalfe's Law is based on the assertion that interaction between two nodes is of value, thus more nodes, more potential interactions, hence the value of the network increases.

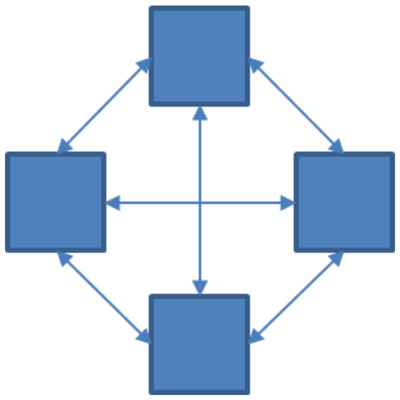

Figure 2: Multiple Nodes in a Network Illustrating Potential Interactions

In this 4 node network, in the context of Metcalfe's Law, there are 6 potential bidirectional interactions, following the formula

$$Bidirectional\ Interactions = Nodes\,\frac{(Nodes - 1)}{2}$$

When viewed in a security context, however, the interactions all become security vulnerabilities which need rules to govern them. Metcalfe's Law stops at generic interactions; however, a security model must take into account the types of interactions, such as the commonplace operations *read*, *write*, and *execute*. Expanding the rule of thumb to include the types of operations doesn't change the order of the equation, but it definitely scales it up, as shown:

$$Security\ Interactions = Nodes\,(Nodes - 1) \times (Operations)$$

Returning to the example of Figure 2, with the assumption of three basic types of operations (read, write, execute), the number of security interactions is 36. Granted, modern requirements management, programming, and validation techniques could formally validate every path in this system; however, a modern software system built upon modular software, shared libraries, all manner of message interfaces, and numerous network connections becomes rapidly unmanageable. A system of 100 nodes (with the same three basic operations of read, write, execute) presents a significantly larger number of security interactions: 29,700.

The counter-point, and it's worth noting, is that there will only be a few labels of interest (e.g. unclassified, secret, etc.) applied across all the nodes, so really a verification needs to be concerned only with the interaction amongst classes, not individual nodes of a particular class. The key detail here is that membership of a class (e.g. a node is unclassified) must be completely trusted to not change (e.g. maliciously change to secret) in order for class-based verification to be successful. In a typical Security Kernel context, unfortunately, this is a house of cards. Compromise of a single node, assumed to be in particular class, compromises all other members of the same class, and can even lead to a node maliciously changing its class to access and compromise other classes. Nevertheless, the idea is sound, but requires a different construct – the separation kernel.

MULTIPLE INDEPENDENT LEVELS OF SECURITY USING SEPARATION KERNELS

Separating security domains in modern complex systems is a straight-forward approach – simply use a separate set of computers, interfaces, and networks for each. This is a great way to simplify the problem, but it's atrocious when it comes to the practical aspects of Size, Weight, Power and Cost. The advent of the World Wide Web drove explosive growth of website hosting companies, who quickly clamored for a way to host multiple customer specific webservers on a common pool of shared hardware, rather than separate physical units for each customer. New hardware sharing technology, called *Virtualization*, allowed web-hosting companies to install and manage virtual machines for individual customers across a large pool of hardware. It's in this technology that we find the first key building block for a SWaP-C optimized MILS Network Reference Architecture.

The many details and variants of virtualization technologies are beyond the scope of this paper as there is one and only one type of virtualization which is applicable to MILS – the Type I or *bare-metal* hypervisor. This type of hypervisor separates the resources of the physical processor (e.g. cores, memory, I/O) into explicit sub-set domains, and then provides a virtual machine for *guest operating systems* to run. These guest operating systems are completely separate from each other, and are only allowed to access the resources they have been allocated by the hypervisor, as shown in Figure 3.

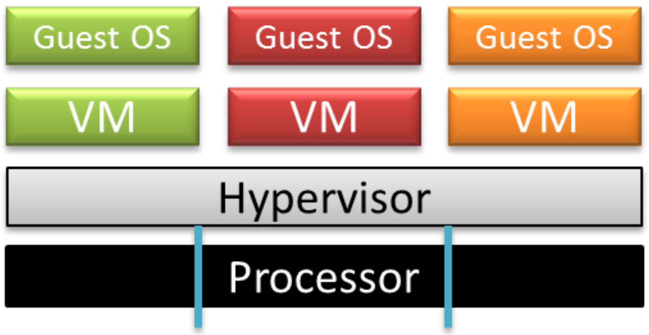

Figure 3: Type I Hypervisor with Three Virtual Machines

The Type I Hypervisor is a separation kernel, and is directly applicable to the creation of MILS architectures, providing a simpler approach to the validation of security interactions across multiple nodes. The Virtual Machines become the method by which to segment nodes into hard-enforced classes of security levels, and the number of security interactions is reduced to a dependency on the number of classes and interclass operations, not nodes.

$$\begin{aligned} &MILS\ Security\ Interactions \\ &\quad = Classes\ (Classes - 1) \\ &\quad \times (Interclass\ Operations) \end{aligned}$$

The difference between the MLS security kernel and MILS separation kernel approaches is illustrated by meetings in conference rooms. In a MLS context, image a group of people, all at different security levels, attempting to carry-on a meeting about a variety of topics at different security levels, while all in the same room. A very complex set of rules and agenda would be needed which governed minimum distances and speaking volumes, note passing and disposal, lines of sight, etc. With only three or four people in the room, it could be done, but extremely difficult.

The easier approach would be to have separate conference rooms assigned to a specific security class. Conversations at a given security context would only be held within the appropriate room, and would be free to proceed without restriction. Instead, interactions between the rooms would be governed by a very small and manageable set of rules, following standard BLP and Biba constructs. A trained and trusted set of runners between the room would follow and enforce those rules. For example, if a person in the secret room attempted to send information to all other rooms, the runner would only deliver the message to the appropriate rooms (e.g. other secret or higher) and would not deliver the message to inappropriate rooms (unclassified). In addition, the runners are themselves guards, not allowing movement of personnel between the rooms.

In an MILS / Hypervisor context, the rooms are the virtual machines, and the trained and trusted runners / guards are the hypervisor, and the shared asset is the building which contains the conference rooms.

A quick criticism of this approach is the potential for a *covert channel* between the rooms which allows inappropriate movement of information. In the building / conference room / runners example, a covert channel between rooms is a real possibility, but it's here were the analogy ceases to be appropriate. A processor with a hypervisor is a significantly more unforgiving and constrained construct than an office building. The proper function of the hypervisor is to ensure that resources are shared appropriately in such a way that each virtual machine is completely unaware of and unaffected by the existence of the other virtual machines. Without this construct, the entire operation of the hypervisor falls apart. Covert channels aren't just security vulnerabilities – they represent a fundamental design and implementation failure of the hypervisor itself to do its essential function. This may seem to be a bit of a hand-wave on "trust a hypervisor", but it should be taken as indication of continual commercial investment and importance in the quality, assurance, and functionality of any hypervisor product.

SECURE COMMUNICATIONS WITH MILS

Another potential vulnerability in the MILS approach is the security of the *data in transit* between virtual machines, whether they are on the same hardware or separate hardware. For this reason, data-in-transit protection is important, but even more importantly it simply extends the concept of the MILS separation kernel out to the network.

In the MLS example, trusted labeling hardware provides a method to mark traffic as part of one class or another, and assumes that data will be received at other nodes by a corresponding set of hardware which follows the rules for the labels. This approach relies on the use of multiple non-standard (not COTS) network interfaces to apply and assess labels on the network traffic.

Rather than using a trusted labeling approach, an alternate approach of a *grey network* was presented by US AFRL [1]. In this approach, double-nested commercially available NSA Suite-B Virtual Private Network (VPN) clients are used to provide data-in-transit encryption across the network, in accordance with the Commercial Systems for Classified (CSfC) program, as shown in Figure 4.

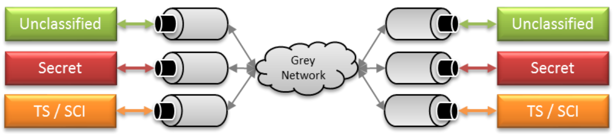

Figure 4: Grey Network using nested Virtual Private Networks

This approach uses a common network through which all data is transported. All data, regardless of classification, is encrypted via an inner VPN and an outer VPN. In this way, the network cannot inadvertently mix or mirror data in-the-clear, and snooping of the data would require the ability to decrypt both layers of VPN.

The previous AFRL work demonstrated a significant number of benefits to this approach with regard to total cost of ownership [2] by slashing the need for separate hardware and network assets for each classification domain. For example, the use of a grey network for what would have been two separate networks cuts the SWaP-C requirement for central network gear in-half while using standard COTS equipment.

From a functional standpoint, this means each guest operating system in a virtual machine must establish a network connection via a VPN to a peer over existing connections. The Hypervisor itself can provide end-points for the outer VPN, ensuring that processes or users within a particular class do not attempt connection to inappropriate peers. Using additional Virtual Machines to establish, route, and secure connections, as shown in Figure 5, adds additional encapsulation and separation to the entire architecture, which is important for validation.

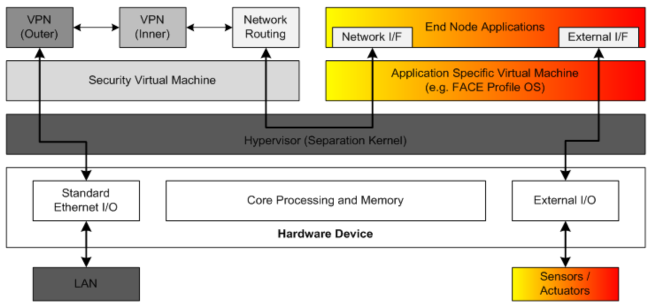

Figure 5: MILS Communication using Separate Communication Security Virtual Machine

MILS NETWORK REFERENCE ARCHITECTURE

The MILS Network Reference Architecture utilizes two fundamental technologies/approaches: Hypervisors as separation kernels, and CSfC-style VPN nesting for data-in-transit security.

A Network Centric Reference Architecture was presented previously in [3], and subsequently with revisions in [4]. In that previous work, the architecture clearly identified the need for separate and duplicate networks and resources for each level of security. This approach, although simple, straightforward, and low-risk does not meet the modern needs of a SWaP-C constrained environment and the explosion of growth in data. The US Army's VICTORY Architecture [5] also shows a separate set of VICTORY Databuses for each domain. Subsequent work in [6] proposed an evolved diffusion of the VICTORY Databus via software defined networks and big data processing constructs, but noted the information assurance challenge presented by the fluid nature of cloud computing.

The MILS Network Reference Architecture, as shown in Figure 6, presents a modification to the revised architecture shown in [4] to collapse the separated networks into a single MILS network. This is a critical first step to realizing the advantage of the diffused network shown in [6].

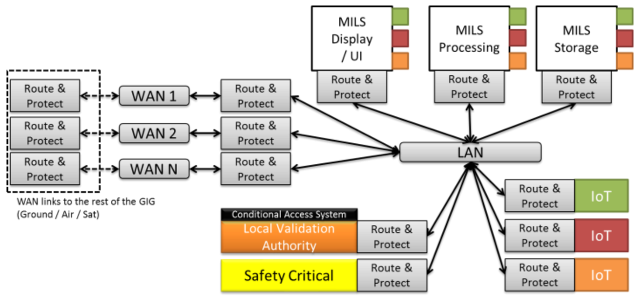
MILS Display / UI
MILS Processing
MILS Storage
Route & Protect
WAN 1
WAN 2
WAN N
LAN
WAN links to the rest of the GIG (Ground / Air / Sat)
Conditional Access System
Local Validation Authority
Safety Critical
IoT

Figure 6: MILS Network Reference Architecture

The MILS Network Reference Architecture envisions various types of network members similar to what was presented in [3], [4], and [5], but now shows the addition *route and protect* functions to all blocks, ensuring that the all nodes on the network are grey and properly routed to peers.

SAFETY AND THE MILS NETWORK REFERENCE ARCHITECTURE

What is not immediately obvious, but is included in the architecture, is the addition of Safety Critical functions and members on the network. Previous work in [7] presented the capability of a modern Ethernet network to provide real-time control in place of previous communication systems used in safety critical applications. As suggested by P. Skentzos of Dornerworks in [8], commercial standards for Safety Critical applications (FAA DO-178C Level E through A) have strong correlation with the Evaluation Assurance Levels (EAL 1 through 7) typically used for security accreditation. At higher levels, both require formal methods of both design and verification. As suggested in [1] and [8], a hypervisor can be designed and implemented in such a way that it can be formally validated. Referring to the early discussion of security interactions, a hypervisor with virtual machines dramatically limits the number of classes (e.g. 3 or 4), and therefore the total number of security interactions which need to – and can – be formally developed and validated.

This is important because it means not only can multiple independent levels of security be mixed on a network, but multiple levels of safety criticality. Formally validated hypervisors can be used in both cases, and methods building upon concepts in [7] are in standardization now for Time Sensitive Networking [9].

RECONFIGURATION AND THE MILS NETWORK REFERENCE ARCHITECTURE

Another critical consequence of the MILS Network Reference Architecture is reconfiguration on user-level or role-based authentication. With shared hardware and network assets, it is now possible to dynamically reconfigure what is and is not allowed on a portion of the network and computing resources based on a user's credentials, as shown in Figure 7.

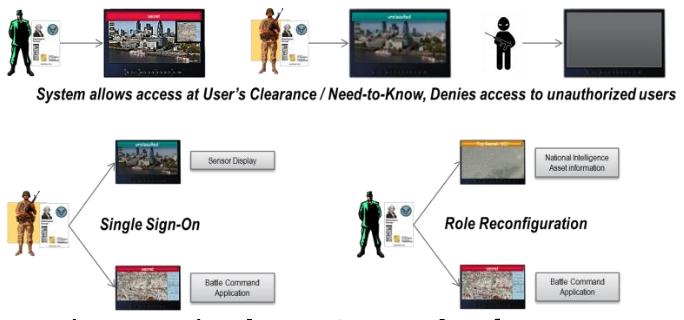
System allows access at User's Clearance / Need-to-Know, Denies access to unauthorized users
Sensor Display
Single Sign-On
Battle Command Application
National Intelligence Asset information
Role Reconfiguration
Battle Command Application

Figure 7: Using the MILS Network Reference Architecture for Reconfiguration

Hypervisors can deploy, suspend, or replace various virtual machines based on the current user context, as shown in Figure 8. Virtual machines can even flow across network resources to run on other hardware (e.g. a different display in a vehicle) as needed.

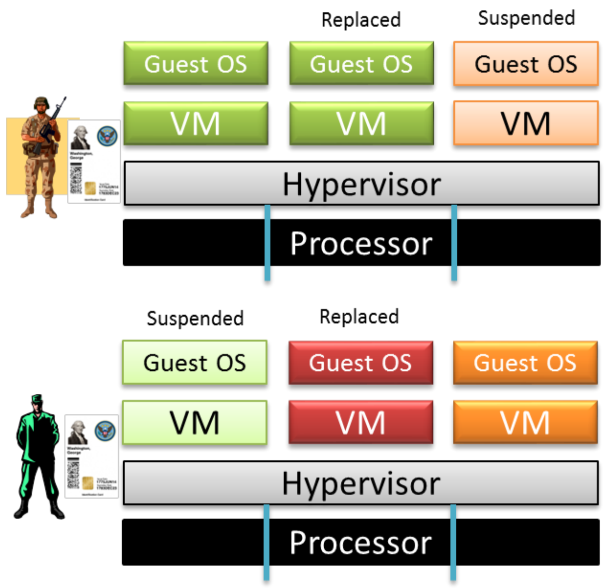

Figure 8: Reconfiguration of Virtual Machines based on User

Virtual Machine images can be protected with standard Data-At-Rest techniques, and critical program information can be protected as need via techniques beyond the scope of this paper or forum. Given these methods, however, secure systems can protect themselves from unauthorized use if compromised. Additional details on in-field authentication mechanisms are presented in [10].

SWAP-C OPTIMIZATION AND THE MILS NETWORK REFERENCE ARCHITECTURE

Of significant importance to the use of the MILS Network Reference Architecture is SWaP-C optimization. In some cases, the savings assume direct elimination of redundant hardware. In other cases (e.g. storage), total aggregate capacity is maintained, but individual unit overheads are eliminated. Table 1 shows an example of the potential SWaP-C savings on a vehicle using the MILS Network Reference Architecture.

Table 1: SWaP-C Comparison for Three Domains

Element	Separate	MILS	SWaP-C savings
Displays	3	1	66%
Mission Processors	3	1	50%
Storage	3	1	25%
Network Switches	3	1	50%

Additional devices, such as device interfaces (*Internet of Things* style connections) may be able to provide some reduction as they can serve multiple domains at once rather than needing duplicate units.

DEVELOPMENT AND VALDIATION IMPACTS

One of the most difficult aspects of a secure system is the development and validation (or accreditation) process. A strong benefit of the MILS Network Reference Architecture is the ability to encapsulate and separate development and accreditation tasks with strong boundaries. Take the example of a sophisticated IR camera system with dual use on a platform: Active Protection System and Driver's Vision Enhancement. In a traditional approach, two separate camera systems would be used with each tailored to the specific needs and classification of its parent system.

If, on the other hand, a single camera with the highest capability is provided – assume the APS needs are higher than that of DVE – it can be difficult to connect those system as they have differing security (and safety) contexts. Suppose the APS capabilities (e.g. resolution) are classified, and the image analysis algorithms are deemed critical program information. On the other hand, the DVE system is intended for use by personnel without clearance (the driver). The MILS Network Architecture provides a solution to use the same camera for both applications, and to develop and accredit in an isolated manner, as shown in Figure 9.

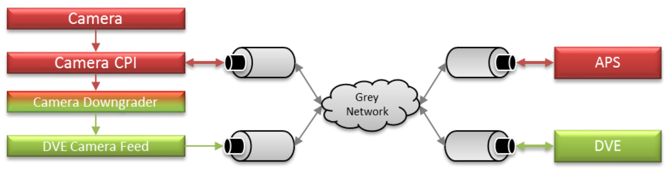

Figure 9: Using the MILS Network Reference Architecture for Design and Accreditation Encapsulation

In this example, each of the colored boxes, except for the *camera* is assumed to be virtualized. The Camera CPI runs in a single VM, and is tightly defined to do two things: receive / process the native camera data and to pass it to another functional block. In this case, it provides it over the grey network to an APS VM plus it provides it via the local hypervisors (or via grey network) to a purpose built cross-domain camera downgrader. The camera downgrader is a purpose built application running in a single VM which is designed to do two things: downgrade received imagery to a lower specification to pass to another functional block. The DVE camera feed is just another VM which receives camera data from an input and sends it to the DVE system over the grey network.

The benefit here is that each of these blocks is simply defined with a small number of security interactions. The camera CPI can be developed, accredited, protected, and deployed without any concern for the demands of the DVE system or any concept of cross-domain. Likewise, the camera downgrader can be developed, accredited, protected, and deployed without any specific knowledge of or interaction with Camera CPI. It could be designed for a wide range of input capabilities, such that upon inspection by an adversary, it is determined that it can downgrade input resolutions which lay somewhere between unclassified commercially available and the maximum possible given the data path bandwidth, but isn't specific about the actual classified input resolution. Similarly, the DVE Camera Feed application within a separate Virtual Machine has no correlation at all with the actual camera capabilities, as it is only receiving the downgrade feed from the APS camera.

By encapsulating each of these to small functional blocks each within their own Virtual Machines, the modern benefits of encapsulation, service oriented architectures, and abstraction can be gained while maintaining strong security boundaries for both development and accreditation.

CONCLUSION

The MILS Network Reference Architecture is a natural evolution of previous network reference architecture approaches. It leverages the strong commercially driven technologies of the broader information technology world for greater flexibility and security in systems. Foundational work already performed by AFRL for fixed install is immediately applicable to embedded application for SWaP-C sensitive applications. In addition the MILS Network Reference Architecture opens multiple additional use cases for safety, reconfiguration, and lower risk development and acredidation.

REFERENCES

[1] "SecureView Overview," United States Air Force Research Laboratory, October 7, 2013, available from http://www.ainfosec.com/wp-content/uploads/2013/10/SecureView_Overview_Master_PA_Cleared_7Oct13.pdf

[2] R. Durante, J. Woodruff, "SecureView: Government/Industry Collaboration Delivers Improved Levels of Security, Performance, and Cost Savings for Mission-Critical Applications," United State Air Force Research Laboratory, December 17, 2012, available from http://www.ainfosec.com/wp-content/uploads/2013/02/SecureView.pdf

[3] D. Jedynak, M. Macpherson, and B. Dolbin, "Ground Combat Systems – Common Vehicle Electronics Architecture and Application," Proceedings of the 2010

Ground Vehicle Systems Engineering and Technology Symposium, Dearborn Michigan, August 2010

[4] D. Jedynak, “Benefits of Intra-vehicle Distributed Network Reference Architecture,” Proceedings of the 2011 Ground Vehicle Systems Engineering and Technology Symposium, Dearborn Michigan, August 2011

[5] M. Moore, K. Saylor, J. Allardyce, R. Serenelli, W. Mononey, and M. Munson, “VICTORY 101 – An Introduction and Technical Overview,” Proceedings of the 2011 Ground Vehicle Systems Engineering and Technology Symposium, Dearborn Michigan, August 2011

[6] D. Jedynak, “Beyond VICTORY – Cloud Computing in Military Vehicles,” Proceedings of the 2013 Ground Vehicle Systems Engineering and Technology Symposium, Troy Michigan, August 2013

[7] D. Jedynak, “Open Standard Approach for Real Time Control over Ethernet,” Proceedings of the 2012 Ground Vehicle Systems Engineering and Technology Symposium, Dearborn Michigan, August 2012

[8] P. Skentzos, “Software Safety and Security Best Practices: A Case Study from Aerospace,” Proceedings of the 2014 Ground Vehicle Systems Engineering and Technology Symposium, Novi Michigan, August 2014

[9] IEEE 802.1 Time-Sensitive Networking Task Group, available from http://www.ieee802.org/1/pages/tsn.html

[10] D. Jedynak, “Subscriber Conditional Access Solution for Dynamic In-Field PKI Authentication,” Proceedings of the 2015 Ground Vehicle Systems Engineering and Technology Symposium, Novi Michigan, August 2015