A SECURE AND TRUSTED BOOT PROCESS FOR AVIONICS WIRELESS NETWORKS

Konstantinos Markantonakis, Raja Naeem Akram, Royal Holloway, University of London. Egham, UK

Abstract

Integrated Modular Avionics (IMA) in existing deployments is a collection of inter-connected avionics equipment supported by wired technology, with stringent reliability and safety requirements. If the inter-connecting wires are physically secured so that a malicious user cannot access them directly, then this simplifies the security management of the network. However, substituting the wired network with a wireless network – also referred to as an Avionics Wireless Network (AWN) – brings a set of new challenges related to assurance, reliability, and security; even for a specific set of well-defined and non-critical tasks. The AWN has to ensure that it provides at a minimum the existing required levels of safety offered by the equivalent wired network. These challenges are underpinned by a necessity to boot the AWN to a secure and trusted state, before it can be used to bridge different parts of the IMA in an aircraft. In this paper, we discuss the security and trust challenges an AWN boot process might face, along with highlighting a potential solution. Finally, the paper evaluates the proposed validation solution that meets the stated security requirements, based on the security challenges discussed.

Introduction

In existing digital avionics, individual devices are linked via wired connections. A potential alternative to the wired network is wireless, specifically an Avionics Wireless Network (AWN) [1]. In a restricted network environment, where all participants are vetted (offline) beforehand, managing security and trust is relatively easy in comparison to a dynamic network where prior vetting becomes impractical. Therefore, challenges of security and trust are unique in some respects in the avionics industry. In addition, a failure in security and trust in avionics industry might have severe consequences. Therefore, for an AWN we require (and recommend) robust technological mechanisms to assess and associate trust with individual entities. In this paper, the trust established using technological means, in whole or in part, is referred to as “digital trust”.

Similar to the diversification of computer systems, digital trust also comes in several different incarnations. Each computing domain has defined and articulated the notion of digital trust in a specific manner that satisfies its requirements. Therefore, the assumption that digital trust will have a single definition is difficult to substantiate. Individual definitions of digital trust might be valid in their respective domains. However, issues arise when a definition of trust in one domain is applied to a different domain without adequately adjusting it, as for example, in the notion of digital trust related to computer security via provenance [2], [3]. Such concerns are rare but nonetheless exist.

In the field of information security, the measurement and validation of digital trust and trustworthiness play crucial roles. The foundation of secure and trusted computing[1] can be argued to be based on the effectiveness of digital trust evaluation and validation mechanisms [8]. In such a reliability-critical and security-sensitive environment as an AWN, verification/validation of digital trust is crucial.

Digital Trust

The definition of trust, taken from Merriam Webster’s online dictionary[2] is a “belief that someone or something is reliable, good, honest, effective, etc."

Based on this, we generically define digital trust as “a trust based either on past experience or evidence

[1] Secure and Trusted Computing: This term refers to the efforts made toward enabling technologies to ascertain trust in a device’s state and security. For example, Trusted Computing Group (TCG) [4], [5], ARM TrustZone [6] and M-Shield [7].

[2] Website: http://www.merriam-webster.com/dictionary/trust

that an entity has behaved and/or will behave in accordance with its self-stated behaviour." The self-stated purpose of intent is provided by the entity and this may have been verified/attested by a third party or trusted manufacturer. The proof that the entity satisfies the self-stated behaviour can either be gained through past interactions (experience) or based on some (hard) evidence like validatable/verifiable properties certified by a reputable third party (i.e. Common Criteria evaluation for secure hardware [9]) or based on strong business relationships and non-digital trust in the manufacturer. This definition is not claimed to be a comprehensive definition for digital trust that encompasses all of its facets. However, this generic definition will be used as a point of discussion for the rest of the paper.

Elements of Trusted Boot

The core idea is that every part of an AWN boots to a secure state when the network is powered-on. The AWN can then provide an assurance to the aircraft systems that it has securely booted to a trustworthy and reliable state – before it transits to an operational state to serve them. The boot process for trust verification measures the integrity (or any other properties) of the succeeding individual elements in this process, before transferring control to them. For the concept of trusted boot, there are three potential variants.

Secure Boot: Secure boot is a security validation during the boot process that ensures that a component can only be loaded if the configuration of the succeeding component is verified. If a modification is detected, the bootstrap process is interrupted.

Authenticated Boot: Authenticated boot is a process that ensures that remote parties can verify the properties (i.e. integrity values) of each of the components involved in the boot process – the boot configuration.

Trusted Boot: Trusted boot is a combination of both the secure and the authenticated boot. The trusted boot process measures certain properties of the succeeding boot component (in the boot configuration) and if the properties do not satisfy the security requirement, it terminates the boot process. In addition, the trusted boot process can provide a validation to a third party about its trusted state, when requested.

In this paper we are mainly concerned with the trusted boot process, in which all elements of an AWN node boot up to a secure and validatable state. In subsequent sections, we first look into how a trusted boot works based on the Trusted Platform Module (TPM) and how applications can securely execute in the Trusted Execution Environment (TEE), before we investigate the three variants of providing trusted boot mechanisms for an entire AWN.

Secure and Trusted Computing

In the real world, trust in an entity is based on a feature, property or association that is entailed in it. In the computing world, establishing trust in a distributed environment also follows the same assumptions. The concept of trusted platforms is based on the existence of a trusted and reliable device that provides evidence of the state of a given system. How this evidence is interpreted is dependent on the requesting entity. Trust in this context can be defined as an expectation that the state of a system is as it is considered to be: secure. This definition requires a trusted and reliable entity called a Trusted Platform Module (TPM) to provide trustworthy evidence regarding the state of a system. Therefore, a TPM is a reporting agent (witness), not an evaluator or enforcer of the security policies. It provides a root of trust on which an inquisitor relies for the validation of the current state of a system.

The TPM specifications are maintained and developed by an international standards group called the Trusted Computing Group (TCG)[3] Today, TCG not only publishes the TPM specifications but also the Mobile Trusted Module (MTM), Trusted Multi-tenant Infrastructure, and Trusted Network Connect (TNC). With emerging technologies, service architectures, and computing platforms, TCG is adapting to the challenges presented by them.

Trusted Platform Framework

The basic framework for the trusted platform is to have a root of trust (preferably in the hardware) and

[3]Trusted Computing Group (TCG) is the culmination of industrial efforts that included the Trusted Computing Platform Association (TCPA), Microsoft's Palladium, later called Next Generation Computing Base (NGSCB), and Intel's LaGrande. All of them proposed how to ascertain trust in a device's state in a distributed environment. These efforts were combined in the TCG specification that resulted in the proposal of TPM.

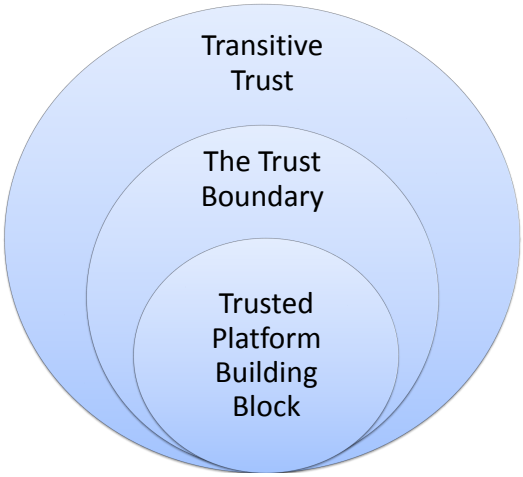

Figure 1. Trusted Platform Framework [10]

trust in it is necessary if an entity has to measure the trustworthiness of a system. The root of trust in the TCG specifications [4], [5] combines Root of Trust for Measurement (RTM), Root of Trust for Storage (RTS), and Root of Trust for Reporting (RTR). The RTM is an independent computing platform that has a minimum set of instructions, which are considered to be trusted for measuring the integrity matrix of a system. On a typical desktop computer, the RTM will be part of the BIOS (Basic Input Output System) and in this scenario, it is referred to as the Core Root of Trust for Measurement (CRTM). Where the RTS and RTR are based on an independent, self-sufficient, and reliable computing device that has a pre-defined set of instructions to provide authentication and attestation functionality, such a device is referred to as a Trusted Platform Module (TPM).

A platform can be considered a trusted platform if it has a TPM and supporting architecture for the "Trusted Building Block" (TBB). The TBB includes a CRTM, a physical connection between the CRTM and the motherboard (of the platform), a connection between the TPM and the motherboard, and functionality to detect physical presence. Physical presence implies the direct interaction of a user with the platform, which is traditionally based on a secret credential that in theory is only known to the user. By verifying the credentials, the platform assumes that the platform owner is physically present. Figure 1 illustrates the trusted platform framework.

The trust boundary is a combination of the TBB and roots of trust. A TPM extends the trust from roots of trust through transitive or inductive trust. A transitive trust is a process that enables a root of trust to provide a trustworthy description (e.g. hash generation) of a second function (e.g. software). The requesting entity can then verify whether it can trust the second function based on the description provided by the relevant TPM. The rationale behind transitive trust is that if an entity trusts the TPM of a platform, it will also trust its measurements.

In this section, the discussion of secure and trusted computing mainly focused on the TPM. There are other proposals for secure and trusted computing but none has the status of the TPM specifications. We will discuss a few of these proposals in later sections, to contrast with the TPM architecture.

A. *Trust and Trustworthiness*

From the discussion in this section, we can delineate two distinct types of trust frameworks: *hard trust* and *soft trust*. The term *hard trust* refers to architectures that base the measurement/foundation of trust on verifiable and independently validated hardware (e.g. TPM [5], ARM TrustZone [6]). In contrast, the term *soft trust* is associated with trust measurement and assessment mechanisms that do not rely on trusted hardware: examples of soft trust can be reputation-, context- and content-based trust mechanisms.

Hybrid trust combines soft and hard trusts to provide a potentially comprehensive approach. In the field of security, a substantial number of trust proposals can be categorised as hard trust. This is not to say that soft trust might not be valid or applicable to the security domain [11], [12]. However, soft trust on its own might not be the best approach to progressing with secure and trusted computing. In the rest of the paper, we discuss hard-trust based mechanisms for secure and trusted computing.

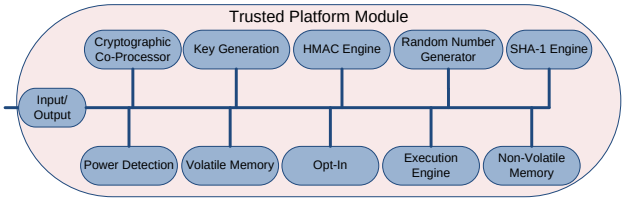

Figure 2. Generic Architecture of Trusted Platform Module [5]

Whether a *soft*, *hard* or *hybrid* trust approach is used, it can be divided into two parts. First is

the trusted measurement and reporting framework and second is the mechanism to generate a score. The generated score will represent the trustworthiness of the relevant entity/information. Data provenance mechanisms can be used to measure and report the state/quality of the data [13], [14]. Based on these measurements and reports, trustworthiness can be calculated; however, data provenance does not become part of the calculations that ascertain the trustworthiness of data. Similarly, TPM is a trusted and secure measurement and reporting hardware system, where the calculation of the trustworthiness of a system is left to the inquirer (i.e. the entity that requests the integrity report from the TPM) [15].

Therefore, security and reliability of the trust measurement and reporting agent are as crucial as the trustworthiness of the system. The basic premise is the invariability and effectiveness of the measurement and reporting mechanism even when in the control of a malicious entity. If a malicious entity can influence the trust measurement and reporting mechanisms then calculation of trustworthiness is of no value. For this reason, hard trust is usually the preferred choice for providing proof that the trust measurement and reporting mechanism are reliable and tamper-resistant, satisfying the requirement for an effective mechanism even when controlled by an active adversary. The calculation of trustworthiness is dependent on the evaluator and it may be independent of the trust architecture — except for mechanisms that integrate hard trust with reputation-based systems [11]. For example, if a malicious user accepts an untrusted system as trusted, then he/she is taking the risk. In such systems a malicious user can still report that system 'A' is untrustworthy even when the trustworthiness of system 'A' is high.

Trust in Security and Privacy

In this section, we briefly discuss the TPM and the Mobile Trusted Module (MTM). Subsequently, we discuss the initial promise of the trusted computing initiative and why in reality it did not get the traction that was expected. Finally, we evaluate the potential future of trusted computing.

Trusted Platform Module

The basic TPM architecture and its different components are shown in Figure 2. For in-depth discussion of individual components and their functionality please refer to [15], [16].

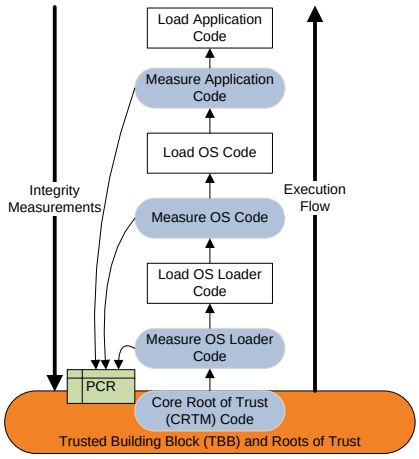

Figure 3. Trusted Platform Boot Sequence

Secure Boot (Measurement Operation): When a user boots up her computer, the first component to power up is the system BIOS (Basic Input/output System). On a trusted platform, the boot sequence is initiated by the Core BIOS (i.e. CRTM) that first measures its own integrity. This measurement is stored in PCR-0[4] and later it is extended to include the integrity measurement of the rest of the BIOS. The Core BIOS then measures the motherboard configuration setting, and this value is stored in PCR-1. After these measurements, the Core BIOS will load the rest of the code of the BIOS.

The BIOS will subsequently measure the integrity of the ROM firmware and ROM firmware configuration, storing them in PCR-2 and PCR-3 respectively. At this stage, the TBB is established and CRTM will proceed with integrity measurement and loading of the Operating System (OS).

The CRTM measures the integrity of the "OS Loader Code," also termed the Initial Program Loader (IPL), and stores the measurement in the PCR. The designated PCR index is left to the discretion of the

[4]Platform Configuration Register (PCR): A Platform Configuration Register (PCR) is a 160-bit (20 bytes) data element that stores the result of the integrity measurement, which is a generated hash of a given component (e.g. BIOS, operating system, or an application). Therefore, a group of PCRs form the integrity matrix. The process of extending PCR values is: $PCR_i = Hash(PCR'_i||X)$, where i is the PCR index, PCR'_i represents the old value stored at index i, and X is the sequence to be included in the PCR value. "||" indicates the concatenation of two data elements in the given order. The starting value of all PCRs is set to zero.

OS. Subsequently, it will execute the "OS Loader Code" and on its successful execution, the TPM will measure the integrity of the "OS Code". After measurement is made and stored, the "OS Code" executes. Finally, the relevant software that initiates its execution will first be subjected to an integrity measurement, and values will be stored in a PCR then sanctioned to execute. This process is shown in Figure 3, which illustrates the execution flow and integrity measurement storage.

By creating a daisy chain of integrity measurements, a TPM provides a trusted and reliable view of the current state of the system. Any software, whether part of an OS or an application, has an integrity measurement stored in a PCR at a particular index. If the value satisfies the requirement of the software or requesting entity, then it can ascertain the trustworthiness of the system or otherwise take action. As discussed before, a TPM does not make any decisions: it only measures, stores, and reports integrity measurements in a secure and reliable manner. When a TPM reports an integrity measurement, it is recommended that it should generate a signature on the value - avoiding replay and man-in-the-middle attacks [5].

Reporting and Attestation Operations: The attestation process, whether initiated by the relevant user/administrator/third-party locally or remotely, involves the generation of a signature using the respective Attestation Identification Key (AIK) on the (associated/requested) PCR values [10]. The signature assures requesters of the validity of the integrity measurement stored in the PCRs. The choice of the AIK and PCR index is dependent on the respective user, platform (OS) or application.

Trust in Execution Environment

In this section we briefly introduce some of the proposals for a secure and trusted application execution and data storage.

ARM TrustZone: Similar to the MTM, the ARM TrustZone also provides the architecture for a trusted platform specifically for mobile devices. The underlying concept is the provision of two virtual processors with hardware-level segregation and access control [6], [17]. This enables the ARM TrustZone to define two execution environments described as Secure world and Normal world. The Secure world executes the security- and privacy-sensitive components of applications and normal execution takes place in the Normal world. The ARM processor manages the switch between the two worlds. The ARM TrustZone is implemented as a security extension to the ARM processors (e.g. ARM1176JZ(F)-S, Cortes-A8, and Cortex-A9 MPCore) [6], which a developer can opt to use if required.

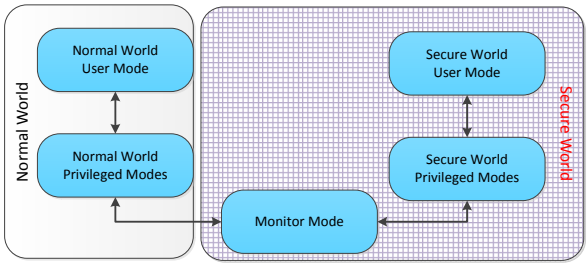

Figure 4. Generic architectural view of ARM TrustZone

GlobalPlatform Trusted Execution Environment (TEE): The TEE is GlobalPlatform's initiative [18]–[20] for mobile phones, set-top boxes, utility meters, and payphones. GlobalPlatform defines a specification for interoperable secure hardware, which is based on GlobalPlatform's experience in the smart card industry. It does not define any particular hardware, which can be based on either a typical secure element or any of the previously discussed tamper-resistant devices. The rationale for discussing the TEE as one of the candidate devices is to provide a complete picture. The underlying ownership of the TEE device still resides with the issuing authority, which is similar to GlobalPlatform's specification for the smart card industry [21].

Trust in the Underlying Hardware

In the early days of computing systems, security was associated with the software. However, recent commercial and economic conditions have forced hardware manufacturers to outsource their production process to countries with cheaper infrastructure costs. While this significantly reduces the integrated circuit (IC) production costs, it also makes it much easier for an attacker to compromise their supply chain and replace ICs with unoriginal or malicious ones. Such items could be counterfeits or hardware Trojans. This threat to the IC supply chain is already a cause

for alarm in some countries [22], [23]. For this reason, some governments have been subsidising a few high-cost local foundries for producing the ICs used in military applications [24]. However, this is not an affordable solution for most of the developing countries.

Counterfeits: Counterfeiting on the global level covers almost everything that is made or manufactured, from spare parts to clothing to prescription drugs. In contrast to other counterfeit items, the ramifications of a counterfeit IC device failure in an electronic system are more than just inconvenience or a minor loss of money. According to [25], the number of counterfeit incidents increased from 3,868 in 2005 to 9,356 in 2008. These incidents can have the following ramifications: (a) original IC providers incur an irrecoverable loss due to the sale of often cheaper counterfeit components; (b) low performance of counterfeit products (that are often of lower quality and/or cheaper older generations of a chip family) affects the overall efficiency of the integrated systems that unintentionally use them, which could in turn harm the reputation of authentic providers; and (c) unreliability of defective devices could render the integrated systems that unknowingly use the parts unreliable, this potentially affects the performance of weapons, airplanes, cars or other crucial applications that use the fake components [26].

Hardware Trojans: Hardware Trojans are malicious circuitry implanted in an IC. The malicious circuit can be inserted for a variety of reasons, including stealing sensitive information, IP reverse engineering or spying on the user. One way of implanting a Trojan into an IC is by compromising the supply chain of ICs and adding a Trojan mask into the original design. Trojan circuits are designed to be very difficult (nearly impossible) to detect by purely functional testing. They are designed to monitor for specific but rare trigger conditions; for instance specific bit patterns on received data or the bus. Once triggered, the actions of the Trojan could be leaking secrets, creating glitches to compromise the security of larger electronic equipment, or simply disabling the circuit. For example, a simple yet deadly Trojan in RSA [27] could be to inject a fault into the CRT inversion step during RSA signature computation that could lead to the compromise of the RSA keys [28].

Countermeasure

Counterfeit ICs and hardware Trojans can be designed to be hard to detect by purely functional testing. However, in the real world ICs also leak information about their internal state unintentionally. This leakage comes via power consumption or electromagnetic emissions caused by varying electric currents flowing through the IC's circuitry. This leakage can be recorded and analysed to detect counterfeits and hardware Trojans. For instance in [29], a gate-level passive hardware characterisation of an IC was proposed to identify defective ICs. However, the gate-level characteristics are dependent on ageing, temperature and supply voltage instability. The authors used the negative bias temperature instability model proposed in [30] to calculate the original characteristics of aged ICs. In another proposal [31], the power consumption of a device was proposed for detecting hardware Trojans implanted in ICs. In this study, process variation noise modelling (constructed using genuine ICs) is used for detecting ICs with Trojan circuits through statistical analysis.

Potential Solutions for Trusted Boot Process in AWN

In this section, we first discuss how trust and trustworthiness can be established in a digital avionics systems and specifically in AWNs. In this section, we will look into how a trusted architecture can be build in an AWN deployment and then how aircraft systems or maintenance crew can verify the status of all the nodes in the AWN.

Trusted Architecture for AWN

As discussed before, the trust starts with the hardware. If there is the potential to have a hardware Trojan in a node, then any security or trust mechanism built on top on this hardware can be vulnerable. Therefore, AWN designers should make sure that the chip fabrication foundry is trustworthy. Even with that provision, they should vet individual chips received back from the foundry to verify that no Trojan was introduced in order to weaken the security of the chip.

After ensuring that individual nodes are manufactured in a trusted fashion, individual nodes should be instantiated using their individual root of trust (TPM). This might generate a set of public key pairs, one for encryption and other for signature scheme.

These key pairs can then be used to provide integrity measurements to an authorised requesting entity in a secure and reliable manner.

The root of trust would provide an assurance that the device is booted to a secure state, as the secure state is known to the requesting entity (or its integrity value, which might be provided to the requesting entity as a manufacturer's signed message or by some other means). After receiving the assurance that the device is booted to a secure/trusted stated, the TEE would provide a strong assurance that during the execution of sensitive parts of the application on individual nodes, no on-board application could alter or interfere with the execution of these sensitive instructions.

AWN nodes can respond to requesting entities' queries about their current state via three different schemes, as follows:

Individual Querying Solution

In this scheme, the requesting entity has to request information from the trusted integrity measurement nodes individually. The pros of this scheme are that the requesting entity does not have to query all the nodes every time an AWN boots up. It can randomly select a subset of the nodes in the AWN and query them. This random selection can also provide a low performance penalty and the AWN can become operational quickly. The downside of such a scheme is that the requester might overlook a node that is not in a secure/trusted state until it is randomly selected – possibly after a long delay. During this period the node might keep on operating in a less secure and/or trusted state.

Peer-to-Peer Querying Solution

In this scheme, there is no centralised entity that requests individual nodes in the AWN to provide their trusted integrity measurements. In peer-to-peer querying schemes, individual nodes query all (communicating) partner nodes for their trusted state. Individual nodes then make the decision whether the node they are going to communicate with in the operation phase is trustworthy or not. If not, then the querying node might raise an error and notify the AWN management node or aircraft system, in order to rectify the problem. The positive aspect of this scheme is that on every power-up for the AWN, states of all nodes are verified. Furthermore, as it is a decentralised scheme all nodes don't have to be individually queried by a centralised entity (i.e. aircraft system). A potential shortcoming of such a scheme is that individual nodes have to store the trusted state of all their partner nodes, to compare with the integrity value they return when queried.

Collective Querying Solution

A collective query scheme enables the centralised requesting entity to only send one request to the AWN. This request is then propagated in the AWN, and all nodes in the AWN then individually respond to the request. A positive aspect of this scheme is that the requesting entity does not have to send individual requests to each node, which might be costly in terms of both computational performance and network communication (load). A potential downside of such a scheme is that all nodes have to generate the integrity measurement (encrypt/sign it) and then send it to requesting entity. This offsets the benefit achieved by sending only a single request over the network. A point to note is that when a node provides an integrity measurement to the request entity, it might include some unique string of bits sent to it by the requesting entity to ensure freshness. Furthermore, the AWN node will then encrypt/sign the integrity measurement and unique string of bits and send them back to the requesting entity. This process is common to all three solutions listed above.

Conclusion

In this paper, we have analysed the requirements for a digital trust mechanism for AWN deployments. Furthermore, we looked into the nature of digital trust and what technological solutions can be used to provide it in the context of an AWN. This discussion was necessary in order to show how digital trust architectures are created in computer systems and how to query them to obtain trust assurances.

We also discussed how these technological solutions can be used in three different deployment scenarios to provide a trusted boot for an AWN. The aim was to provide an assurance to a requesting entity that all nodes in an AWN are booted to a trusted state and the network as whole is secure and trustworthy. Any security protocols that then execute after the boot

process can have a strong assurance that they are running on a device that is in a secure state.

Acknowledgements

The authors acknowledge the support of the UK's innovation agency, InnovateUK, and the contributions of the Secure High-Availability Avionics Wireless Networks (SHAWN) project partners.

Disclaimer

The views and opinions expressed in this article are those of the authors and do not necessarily reflect the position of the SHAWN project or any other organisations associated with this project.

References

[1] R. N. Akram, K. Markantonakis, S. Kariyawasam, S. Ayub, A. Seeam, and R. Atkinson, "Challenges of security and trust in avionics wireless networks", in *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*, IEEE, 2015, 4B1–1.

[2] J. Yao, J. Zhang, S. Chen, C. Wang, D. Levy, and Q. Liu, "A mobile cloud with trusted data provenance services for bioinformatics research", in *Data Provenance and Data Management in eScience*, Springer, 2013, pp. 109–128.

[3] A. Martin, J. Lyle, and C. Namilkuo, "Provenance as a security control", *TaPP. USENIX*, 2012.

[4] *ISO/IEC 11889-1: information technology - trusted platform module - part 1: overview*, English, Online, Standard, International Organization for Standardization (ISO), 2009.

[5] *TPM main: part 1 design principles*, English, Online, Specification, Trusted Computing Group (TCG), 2011.

[6] "ARM security technology: building a secure system using trustzone technology", ARM, White Paper PRD29-GENC-009492C, 2009.

[7] "M-shield mobile security technology: making wireless secure", Texas Instruments, Whilte Paper, 2008.

[8] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing", *Dependable and Secure Computing, IEEE Transactions on*, vol. 1, no. 1, pp. 11–33, 2004.

[9] *Common criteria for information technology security evaluation, part 1: introduction and general model, part 2: security functional requirements, part 3: security assurance requirements,* Common Criteria, 2006. [Online]. Available: http://www.commoncriteriaportal.org/thecc.html.

[10] R. N. Akram, K. Markantonakis, and K. Mayes, "An introduction to the trusted platform module and mobile trusted module", in *Secure Smart Embedded Devices, Platforms and Applications*, Springer New York, 2014, pp. 71–93.

[11] R. Zhou and K. Hwang, "Powertrust: a robust and scalable reputation system for trusted peer-to-peer computing", *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 4, pp. 460–473, Apr. 2007, ISSN: 1045-9219. DOI: 10.1109/TPDS.2007.1021. [Online]. Available: http://dx.doi.org/10.1109/TPDS.2007.1021.

[12] K. Hwang and D. Li, "Trusted cloud computing with secure resources and data coloring", *IEEE Internet Computing*, vol. 14, no. 5, pp. 14–22, 2010, ISSN: 1089-7801. DOI: http://doi.ieeecomputersociety.org/10.1109/MIC.2010.86.

[13] C. Dai, D. Lin, E. Bertino, and M. Kantarcioglu, "An approach to evaluate data trustworthiness based on data provenance.", in *Secure Data Management*, W. Jonker and M. Petkovic, Eds., ser. Lecture Notes in Computer Science, vol. 5159, Springer, Aug. 20, 2008, pp. 82–98, ISBN: 978-3-540-85258-2. [Online]. Available: http://dblp.uni-trier.de/db/conf/sdmw/sdmw2008.html#DaiLBK08.

[14] O. Hartig and J. Zhao, "Using web data provenance for quality assessment.", in *SWPM*, J. Freire, P. Missier, and S. S. Sahoo, Eds., ser. CEUR Workshop Proceedings, vol. 526, CEUR-WS.org, 2009. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.174.3451.

[15] R. N. Akram, K. Markantonakis, and K. Mayes, "Trusted platform module for smart cards", in *6th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, O. Alfandi, Ed., IEEE CS, 2014.

[16] "Trusted computing group, tcg specification architecture overview", The Trusted Computing Group (TCG), Beaverton, Oregon, USA, revision 1.4, 2007. [Online]. Available: http://www.trustedcomputinggroup.org/files/resource_files/AC652DE1-1D09-3519-ADA026A0C05CFAC2/TCG_1_4_Architecture_Overview.pdf.

[17] P. Wilson, A. Frey, T. Mihm, D. Kershaw, and T. Alves, "Implementing embedded security on dual-virtual-cpu systems", in *IEEE Design and Test of Computers*, 2007, pp. 582–591.
[18] , "Globalplatform device: GPD/STIP specification overview", GlobalPlatform, Specification Version 2.3, 2007.
[19] *Globalplatform device technology: device application security management - concepts and description document specification*, English, Online, Specification, GlobalPlatform, 2008.
[20] "Globalplatform device technology: tee system architecture", GlobalPlatform, Specification Version 0.4, 2011.
[21] *GlobalPlatform: GlobalPlatform Card Specification, Version 2.2,* GlobalPlatform, 2006.
[22] D. A. R. P. Agency, *Darpa baa06-40, a trust for integrated circuits*, https : / / www . fbo . gov / index ? s = opportunity & mode = form & id = db4ea611cad3764814b6937fcab2180a & tab = core & _cview=1, Visited, May 2013.
[23] J. I. Lieberman, *The national security aspects of the global migration of the u.s. semiconductor industry*, http : / / www . fas . org / irp / congress / 2003 _ cr/s060503.html, Visited, May 2013.
[24] D. S. B. T. Force, *High performance microchip supply*, http : / / www . acq . osd . mil / dsb / reports / ADA435563.pdf, Visited, May 2013.
[25] U. D. O. Commerce, "Defense industrial base assessment: Counterfeit electronics", Bureau of Industry and Security, Office of Technology Evaluation, Tech. Rep., January, 2010, http : / / www . bis . doc . gov / defenseindustrialbaseprograms / osies / defmarketresearchrpts/final_counterfeit_electronics_ report.pdf.
[26] F. Koushanfar, A.-R. Sadeghi, and H. Seudie, "EDA for secure and dependable cybercars: Challenges and opportunities", in *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, 2012, pp. 220–228.
[27] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978.
[28] D. Boneh, R. A. DeMillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults (extended abstract)", in *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, 1997, pp. 37–51.
[29] S. Wei, A. Nahapetian, and M. Potkonjak, "Robust passive hardware metering", in *International Conference on Computer-Aided Design (ICCAD)*, San Jose, California, USA: IEEE, 2011, pp. 802–809.
[30] S. Chakravarthi, A. Krishnan, V. Reddy, C. Machala, and S. Krishnan, "A comprehensive framework for predictive modeling of negative bias temperature instability", in *Reliability Physics Symposium Proceedings, 2004. 42nd Annual. 2004 IEEE International*, 2004, pp. 273–282.
[31] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using ic fingerprinting", in *Security and Privacy, 2007. SP '07. IEEE Symposium on*, 2007, pp. 296–310.