Security Foundation for a Distributed Cabin Core Architecture

Nicolai Kuntze and Carsten Rudolph
Fraunhofer SIT (Germany), {kuntze|rudolphc}@sit.fraunhofer.de
Oliver Hanka
Airbus Group, Germany, oliver.hanka@eads.net

***Abstract*—Aircraft cabin networks support various functionalities, from entertainment to safety-critical functions such as passenger announcements or light control. For cost, efficiency, energy-reduction and weight architectures for cabin networks should be optimized with respect to required resources and cable in the cabin. This motivates multi-domain architectures. The approach presented in this paper uses hardware-based security to realize a secure efficient multi-domain architecture for cabin networks.**

I. Introduction

In today's aircraft cabins, multiple functions are deployed in order to ensure the safety of the passengers as well as increasing the comfort during the flight. These functions (e.g., cabin lights, passenger announcements, in–flight–entertainment) require a communication network in order to distribute data between the purpose build end–devices as well as their management applications. In modern aircrafts, like the Airbus A350, the network is laid out in a centralized architecture. Multiple switches provide connectivity for the end–devices and are concentrated at a centralized management server [1].

The switches on the lines reaching into the aircraft's cabin, thereby, are usually equipped with powerful computational elements (i.e., CPUs). The purpose of the CPUs is to support automated topology discoveries as well as configuration tasks for the switch elements and possibly attached end–devices. After the initial discovery and configuration phase, however, the service processors in the switch elements are not used and are idle until the next reconfiguration phase. Due to regulatory requirements, these reconfiguration phases are only allowed to occur while the aircraft is on the ground and not during any other operational phase (e.g., taxiing, take–off, cruise, etc.). This means, the capability of these service processors is not used during the major part of an aircraft's life–span.

In order to better utilize the capabilities of those switches and their processing units, distributed cabin architectures have been proposed and evaluated (e.g., [2], [3]). Besides taking advantage of the existing processing power, a distributed architecture has further benefits over a centralized one. The major drawback of any centralized architecture is the single–point–of–failure. In case, the cabin management server or the switching component which acts as the hub of the star topology fails, the whole cabin network is rendered dysfunctional. In order to overcome this problem, redundancy mechanisms are deployed in today's architectures. These, however, require the costly (financial, power consumption and weight wise) deployment of backup components which need to be synchronized with the active system.

A distributed architecture can be designed with integrated redundancy mechanisms which do not require additional hardware. A major challenge for distributed architectures, however, is the security aspect and the trust among peers within the network. To the best of our knowledge, this aspect hasn't been addressed by any researchers for distributed cabin networks so far.

In this paper, we give an insight on the application area of an distributed cabin architecture and outline the security requirements arising from the distributed nature as well as the special avionics use case. Based on these requirements, we introduce a security architecture which is capable of providing trust between peers in a distributed cabin architecture. This *security foundation* can then also be used to build additional security functionality on top, catering for security requirements of the cabin management applications.

The remainder of this paper is structured as follows: First, the vision of a multi-domain cabin network architecture is introduced and motivated. Then, security requirements for such a network are identified and a possible topology is introduced. After explaining the hardware-based security paradigms and security functionalities used in the design, finally, the trust establishment and trust management processes for the different phases of the life-cycle are explained.

II. Vision of a multi–domain distributed cabin architecture

The basic task of a distributed cabin architecture is to provide connectivity between various end–devices within the aircraft's cabin as well as computational capacity in order to execute required managing applications. Figure 1 depicts a logical sample cabin configuration including components required by a distributed cabin architecture.

The figure shows grouped components consisting of two blocks labeled with *S* and *P*. These two blocks represent a switching (*S*) and processing (*P*) element within each functional group. The switching block is used to provide connectivity for end–devices (e.g., handsets, speaker, access points) as well as towards other switching elements within the aircraft's cabin. The processing element hosts the management applications which control the end–devices.

978-1-4799-3383-9/14/$31.00 ©2014 IEEE

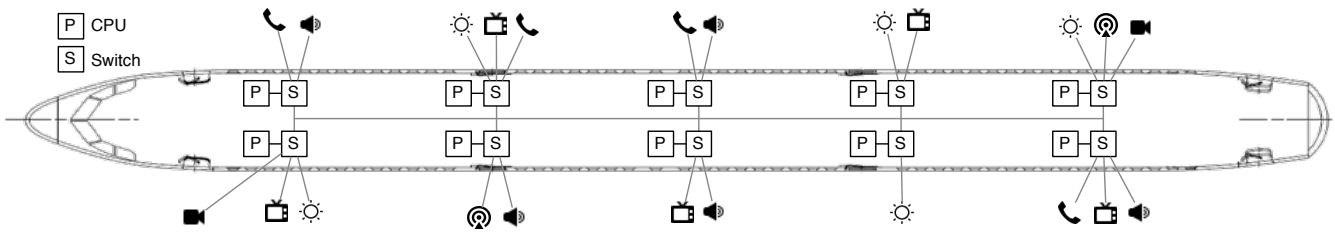

Fig. 1. Logical distributed cabin topology

The cabin network, thereby, interconnects end–devices and management applications which are grouped into different criticality levels. These levels represent the different security requirements of the individual functions as well as the severity of the impact on the aircraft's behaviour in case the functionality is disrupted[1].

End–devices of different criticality levels are required to coexists—without disturbing each other—within the aircraft's network. This means that the aircraft cabin's network must logically separate the various criticality levels. In order to organize these levels, the functionalities and end-devices are assigned to one of the following three security domains, according to [5]:

- *Aircraft Control Domain (ACD)*
- *Airline Information Service Domain (AISD)*
- *Passenger Information and Entertainment Service Domain (PIESD)*

The ACD, thereby, represents the highest (i.e., most critical) domain. Functions within this domain may have a severe impact on the aircraft operation in case they fail. The PIESD, on the other hand, has a negligible safety impact. A malfunction of PIESD applications, however, might lead to a major commercial impact (e.g., compensation for lack of IFE on a long haul flight).

Additionally to connecting end–devices and keeping a logical separation of functions, the distributed cabin network must provide enough computational power and flexibility to host various management applications in a resilient/fail-safe manner. This means that limited hardware failures or connectivity problems should not interrupt the cabin architecture's services. In case a set of processing units fails, other processing units must dynamically take over the execution of the applications, which were previously hosted on the failed nodes. The applications, thereby, range from computationally wise less demanding ones (e.g., passenger call registration) to high demanding applications with on–the–fly video trans-coding, for example.

III. Requirements to avionics network infrastructure

Devices used for cabin networks and the network infrastructure itself need to satisfy a large number of requirements. Many of these requirements are already given by existing safety certification and other constraints for aircraft development. A distributed architecture with redundancy in resources (for computation as well as communication) as well as for communication can provide more efficient ways to achieve a resilient network that still provides core functionality even when network nodes (e.g. routers) are malfunctioning or have been attacked by a malicious intruder. This section focuses on those requirements for which the proposed architecture can make a difference. The challenges to the cabin network in the presented use case of distributed service provisioning mainly impact the stability and resilience of transport, i.e., routing, and several safety and security issues.

a) Requirements concerning routing: All devices in the network need to provide secure identification towards their peer nodes in the network, as these identities are used as the basis for routing in the network. Active devices that do not belong to the original configuration of the network need to be detected and excluded from all communication. Different domains operate within one single network and also functionalities can be distributed over the network. Thus, different types of traffic in the network need to be distinguished and the routing must be aware of the routing issues on the different layers. Routing also needs to consider priorities for different types of messages. Redundancy in the architecture enables reaction to failures. The network must support a (predefined) re-distribution of functions in case of a failure in some elements of the network.

b) Specific safety requirements: The network is used for different types of functionality with different relevance for safety. One example for relevant functionality are passenger announcements, while the entertainment system is not safety critical. The architecture must ensure operability of all safety-relevant functions. Keeping up quality of service for other functions must not block or delay messages for safety-relevant domains. Reactions on security issues must not result in stopping safety-relevant functionality. No components should be able to do a self "log-out" from the architecture and thus blocking functions without providing the functionality via other redundant parts of the architecture.

c) Specific security requirements: As in all (also in current) multi-domain architectures, a proper domain separation is essential. Messages crossing domains (e.g. interfering with light control or passenger announcements from the entertainment domain) must be prevented. Also, flow from the cabin network to other networks in the aircraft must be prevented. The distributed architecture requires that different peer nodes need to

[1]Please note that security and safety are closely related entities which influence each other. Please refer to [4] for a more detailed distinction.

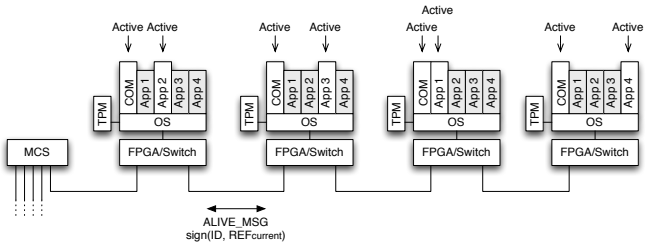

Fig. 2. Topology of a distributed cabin network architecture

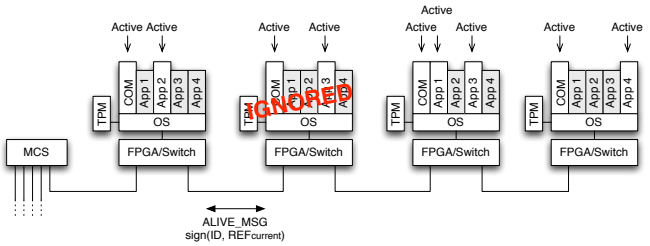

Fig. 3. Malicious service integrated in device

establish trust relations. Nodes need to identify and authenticate each other and also mutual health checks of nodes in the neighbourhood are necessary to build a reliable and secure distributed architecture. Devices in the cabin are in principle accessible by passengers and staff within the aircraft. Therefore, various threats and attack vectors for these networks exist. It must be ensured that attackers cannot take control of network nodes and cannot attach devices to the network that themselves appear as proper network nodes and thus interfering with the correct operation of the network. By protecting network nodes, attack possibilities are pushed to the edge of the network and distributed attack recognition can be implemented.

Certification requirements in the avionics domain also have consequences on the implementation of security functionalities in the devices. From the perspective of security most important is to notice that equipment applied in avionics need to be identical until the actual installation in the target infrastructure. Aside of the MAC address, all configuration and crypto key material need to be identical. Therefore, typical Hardware Security Modules with pre-established secure identities and protocols relying on these identities cannot be applied here without adaptations reflecting this requirement.

IV. Next Generation Cabin Infrastructure

A topology answering to the use case and requirements

- Equal peers (CPU of the FPGA/Switch)
- Functions distributed over all peers
- Activated and standby functions on each peer
- ALIVE messages among the peers (routing and security purpose)

Such an architecture distributes functionalities between the nodes involved. To ensure availability of the overall functionality it is vital to detect and mitigate situations impacting the trust in the individual device. In the architecture described routing hardware or the service provided by the individual device may be compromised or defect. This requires a specific mitigation.

Figure 3 depicts the intended reaction to a malicious service in the network. Each node is able to verify the health state of the service and can initiate a procedure to select a new node to provide the service.

The mitigation of a hardware failure is shown in Figure 4.

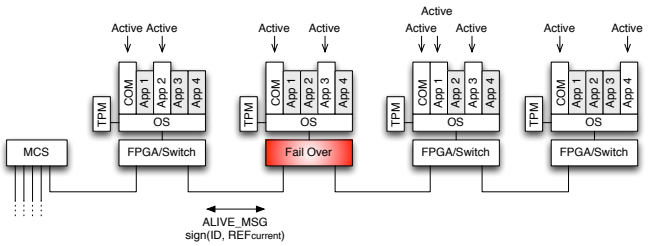

Fig. 4. Failure in hardware

V. Basics of Hardware Security Modules

According to the mission statement of the Trusted Computing Group[2], Trusted Computing based on hardware root of trust has been developed by industry to protect computing infrastructure and billions of end points. The Trusted Platform Module provides the core security functions and serves as a root of trust for each individual device.

TCG defines the TPM in in its available specification 1.2 but also provides the next generation in form of the 2.0 version. Both versions of the TPM design share underlying principles regarding the functionality and functions provided.

The TPM is regarded as a trust anchor bound to an individual system. Trust is defined within the TCG to convey an expectation of behaviour. Hereby it needs to be emphasized that a predictable behaviour does not constitute behaviour that is secure or worth to be trusted. However, various security properties for platforms can be satisfied and controlled based on the TPM. For example, to determine the trust in a certain platform, it is required to identify the identity of the platform. The TPM provides a unique identity for a platform which can either to directly identify a concrete platform or for pseudonymous identification.

The next sections gives a more detailed presentation of how a TPM is used to implement his dedicated role as a root of trust. Specifically, it is shown how different roots of trust in a system design complement each other to build platforms with particular security properties.

A. Roots of Trust

The high level concept of Trusted Computing as defined by the TCG introduces different roots of trust in the system design providing complementary security functions. To attest

[2]http://www.trustedcomputinggroup.org/

on the health of a system each software component needs to be measured [3] beginning form the initialization of the device. In the reference design according to TCG, the initial start of a system begins with the **Root of Trust for Measurement (RTM)**. A RTM measures itself and is implemented using platform features to ensure the tamper resistance of the RTM. The RTM measures the next stage in the boot process and transfers control to it. By this, the RTM already behaves according to the *measure before execute approach* underpinning the overall measurement of the boot process and of software subsequently started on the system.

The second component is the **Root of Trust for Storage (RTS)**, which is usually implemented by protected non-volatile storage within the TPM hardware. The RTS mainly has three roles. First it needs to provide secure storage for the cryptographic identity of the TPM. Second, it also provides secure storage for the keys that are used to encrypt data to be securely stored on (insecure) storage media on the platform or outside the platform. (e.g. symmetric keys for bulk encryption). Finally, the third part includes particular protected registers used to store the measurement information as hash-values (i.e. chains of hash values represented by the final value in the chain.). These registers are called Platform Configuration Registers.

The final root of trust is a securely stored cryptographic key (Endorsement Key EK), the **Roof of Trust for Reporting (RTR)**. This key is used to create attestation identity keys (AIKs) that are then used to digitally sign PCR content in the TPM_Quote command and to certify non-migrateable keys generated by the TPM. AIKs can be used to identify a platform with different levels of pseudonymity.

B. Basic Trusted Platform Features

The TPM provides a large set of security functionalities. The secure distribution of vendor-specific keys in deployment scenarios builds on and uses the following functionalities:

Authentication can be easily based on the Endorsement Key EK that is either established in the TPM by the producer of the TPM or implanted to the TPM in a later stage of the production process. Together with a certificate for the EK, this key is used to build subsequent authentication processes.

Attestation in this context describes the process of reliably reporting the platform status. The TPM provides the functionality for secure remote attestation. Thus, remote entities can get digitally signed information on the current content of platform configuration registers. Protocols for remote attestation are defined in the TCG standards.

Protected Location for keys and other data transferred to the platform is provided by the TPM. In the TPM context, the process of encrypting data with a key protected by the TPM and binding this encrypted data to a particular state of the platform (i.e. particular PCR values) is called *Sealing* . Thus, sealed data can only be decrypted when two conditions are satisfied. First, the same TPM needs to be used with the correct key loaded to the TPM (and optional the correct authentication value/password for the key is given) and second, the platform is in the correct state that the PCR values match.

[3]It is literally *measure*, meaning computing a hash for loaded executable code and on other information defining the behaviour of the platform, such as hardware, configuration files, etc.

VI. Infrastructure Trust Establishment and Trust Management

To establish trust according to the definition of the TCG, it is required to cover the life cycle of the product by introducing an architecture for trust establishment through the stages of production, deployment in the aircraft, operation and, finally, decommissioning. The architecture for trust establishment provides in each stage of the life cycle the necessary framework to address the security requirements to avionics network infrastructure. In the following sections the architecture description is given by explaining the individual life cycle steps. The aim hereby is to ensure the integrity of the devices when the individual device is deployed to the aircraft and during operation.

To ensure the integrity of an individual device, the device needs to be provide a reliable identity and means to attest the software of the device [6]. Both aims can be addressed using the TPM as an example COTS component in the design of the system.

A. Production

According to the certification requirements, each device produced needs to be identical to all other devices of the same type and certification. This requirement also disallows for individual identities for the devices produced as an identity manifests itself in a asymmetric key pair. As device specific keys are part of the certification all devices under the same certificate need to have the same key. Therefore it is not possible to distinguish two devices. It still provides proof on the origin of the device e.g. that the component is a genuine Airbus part.

The identity can be embodied in a security chip derived from the well known TPM having the same identity – called an Endorsement Key (EK) – then all other chips for a particular line of devices. The key is protected by the hardware chip so that compromising the key is rather complicated. In case a single chip gets compromised the EK is revealed. A publicly known EK allows for duplicated devices which endangers enrolment security. The identity of already deployed devices is not affected due to the AIK (device specific key) in place which will be shown in the next section.

The information stored on each device is composed of a set of credentials protected by the security chip that allows to verify the origin and type of device. This set of information is composed of the following parts:

- The security chip carries and protects the **Endorsement Key** which is identical on each device of the same type and certification level.
- A vendor issued certificate which can be verified using a vendor certificate e.g. issued by Airbus. This vendor certificate carries the public key of the root key pair. The

vendor issued certificate is certifies the Endorsement Key as the valid key recognized by the vendor.
- Part of the certificate is a descriptive information on the device type resp. device description.
- Software reference

B. Deployment

The devices are until the time of deployment identical to each other and share the same EK for all devices of the same type and certification. Within the operation of the overall system the individual device is integrated in, it is necessary to provide for each device an individual identity. This is required to ensure that communication can be established between devices preventing man in the middle attacks. During deployment a devices is configured for its specific tasks. In this configuration the step of individualization takes place.

To individualize a device a new key is established also protected by the TPM. Using the TPM, the establishment of keys can be achieved by two different basic mechanisms. Externally created keys can be injected to the TPM and securely stored, or the ability of the TPM to create keys and prove their origin is used. A prime candidate as an individual key is the Attestation Identity Key (AIK) created by the TPM and already meant to be used as a replacement identity concealing the EK.

After the individualization of the device by the establishment of an unique key, this key needs to be published to the other devices in the infrastructure communicating with the device. This requires a knowledge of this unique key on the receiver side. To achieve this knowledge a key distribution process either manually or using an onboard infrastructure is required. A manual approach involves an external system e.g. laptop that provides the process executed by a technician. In case of connectivity to the manufacturer, the laptop could establish an online connection and create an AIK certificate directly by the manufacturer. A dedicated device in the onboard infrastructure could provide a similar service using an automated deployment process as provided similar to concepts discussed in [7], [8]. In an automated scenario, the infrastructure uses an internal representation of the manufacturer (even by allowing a back channel to the manufacturer) with the ability to verify the equipment on the basis of the certificates and TPM used. The device is authenticated on the basis of the unique ID and the status of the software installed.

Information generated and stored on each individual device during the deployment of a device is a

- unique key pair identifying the individual device. The public part of the key pair needs to be communicated through the system the device is part of to allow for device identification.
- a public key identifying the device on each other device in the infrastructure

C. Operation

When a device is successfully initialized, the device is primed for normal operation which consists of an initial peering at boot up and periodic integrity checks. During the initial peering trust relations between pairs of devices are established based on the unique identities and the reported health status. Periodic integrity checks are then used to monitor this status over time. To support a maximum of availability, the trust establishment and maintenance needs to be independent of the payload of the connection. This is a major difference to typical security approaches as this solution separates payload and secure channel. One way to address this channel separation issue is to apply digital signatures on the data stream [9], [10].

Information exchanged during initial peering consists therefore out of

- Individual public keys of the devices involved in the pairwise pairing.
- Software references that allow to verify the status of the involved devices. Typically, these are hash values of the binaries executed on the device as well as on the content of involved configuration files. These references are signed by the manufacturer and can be used by the communication partner to confirm the health status.

On the basis of the exchanged software references, each device is able to evaluate the status of devices it is interacting with. Health checks can be performed on various aspects of the device status and interaction. Most important are

- Software references to ensure that only expected software and therefore behaviour is loaded to a specific device e.g. using SWID tags as introduced by [11]
- Opened network ports, either reported by the device to be evaluated or by it's neighbours or PEPs, provide information on the interaction of the device
- Bandwidth and package monitoring by trusted sensors allows for additional information on the device utilization and health.

According to the result of a health check, the device validating the health check result has to determine a suitable reaction to the check. Here the device acts in it's role as a Policy Enforcement point (PEP). As the availability of the infrastructure is of utter importance, each reaction to security events needs to aim to minimize the impact on the operation. From a high level view, the following four categories of incidents can be distinguished:

- **Compliant** to the expected behaviour. In case of TC assessment on the basis of TPM enforced remote attestation the binary measurements of all components are verified and considered as trustworthy. Also other techniques for the end point assessment show no unexpected behaviour. In this case, correct behaviour can be expected and additional connectivity can be allowed, e.g. for maintenance, for distribution of updates.
- **Outdated** software on the device which can be detected either by reported older measurements on the device using Trusted Computing or known behaviour detected by other means that matches an old and deprecated behaviour. This status might indicate vulnerable systems, but does not directly show an attack. The device can continue to operate until a suitable time for updating the software occurs.

However, additional connectivity should be restricted.

- **Typed malicious** software in case of already known and well understood changes to the software on the device being checked. A malicious modified device may still be providing the intended service and could under special circumstances still be used in the network. These circumstances are threat specific protection measures enforced through the infrastructure the device is connected to. This case allows for targeted reactions to the known changes. If appropriate, rebooting the device or network filters can be used to address the modification.
- Previously **unknown** status or behaviour of a device. In this case, the focus is to quarantine the device in order to prevent the spread of a possible malicious infection. It is important to understand that even in quarantine the functionality of a manipulated device can still be provided. Then, the device interactions need to be strictly monitored until the health of the device can be restored. If possible, the device can be disabled or a specific reboot can be started.

Periodic ALIVE_MSG packets trigger health checks between the nodes to maintain the security level gained in the system. Using these ALIVE_MSG packets allows to gather crucial information on the overall system status. The main application of ALIVE_MSG is building a topology view and gathering of routing information. Together with health checks a more differentiated topology view can be build

- Signed software references allows to determine which functionalities are located on the individual devices
- Lack of ALIVE_MSG means hardware failure of a particular device
- Difference in stored and received software references indicates modified software stack which may point to a malicious device
- Forked message can be detected by verifying the messages signature

During operation redistribution of functionalities between devices may occur due to hardware, software or security incidents. The augmented topology in conjunction with health checks performed on the software level allows to determine the functionalities available in the system. Based on these information, individual trusted peers can be selected and functionalities in the system restored by relocating them to this particular trusted devices.

D. Decommissioning

When a device is removed from an airplane, all cryptographic material needs to be destroyed to ensure that the device can not be put back into the airplane again. The device specific key (e.g. AIK) is revoked for this purpose. The revocation resets the device into the pristine state and allows for **re-deployment**. For a **final destruction** of a device, it is required to revoke the EK which invalidates the device certificate and renders the device useless. It is then also not compliant to the specification of the device as the key is not present any more.

VII. Conclusion and Outlook

This paper provided a first concept on the integration of strong, hardware based identities in the area of aviation technology with a special focus on the processes established in this industry. As shown, strong security means allow not only more secure architectures but provide for improvements in the logistics chain and counterfeit protection.

Next steps in research on security augmented equipment for aviation technology is a more detailed analysis of the processes involved and feasibility studies for the technology in real world scenarios.

References

[1] F. Wolfgang, P. Klose, M. Heinisch, and J. Reuter, "Challenges of future cabin networks," in *4th International Workshop on Aircraft System Technologies*. AST2013, 2013.

[2] N. Audsley and M. Burke, "Distributed fault-tolerant avionic systems-a real-time perspective," in *Aerospace Conference, 1998 IEEE*, vol. 4, 1998, pp. 43–60 vol.4.

[3] R. Hammett, "Flight-critical distributed systems: design considerations [avionics]," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 18, no. 6, pp. 30–36, 2003.

[4] U.S. Department of Transportation, "Networked local area networks in aircraft: Safety, security and certification issues, and initial acceptance criteria (phases 1 and 2)," Federal Aviation Administration, Tech. Rep. DOT/FAA/AR-08/31, 2008.

[5] Network infrastructure and security (NIS) workgroup, "ARINC 664p5, aircraft data networks," ARINC, Tech. Rep., 2001.

[6] N. Kuntze, A. Fuchs, and C. Rudolph, "Reliable Identities using off-the-shelf hardware security in MANETs," in *Proceedings of the International Symposium on Trusted Computing and Communications (TrustCom 2009)*, 2009. [Online]. Available: http://sit.sit.fraunhofer.de/smv/publications/download/TrustCom09.pdf

[7] N. Kuntze and C. Rudolph, "On the automatic establishment of security relations for devices," in *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management*. IFIP/IEEE, 2013. [Online]. Available: http://sit.sit.fraunhofer.de/smv/publications/download/ZeroTouchConf-IM2013-paper.pdf

[8] ——, "Secure deployment of smartgrid equipment," in *Power and Energy Society General Meeting, 2013 IEEE*, 2013. [Online]. Available: http://sit.sit.fraunhofer.de/smv/publications/download/PES2013.pdf

[9] N. Kuntze, A. U. Schmidt, and C. Hett, "Non-repudiation in internet telephony," in *New Approaches for Security, Privacy, and Trust in Complex Systems. Proceedings of the IFIP sec2007. Sandton, South Africa 14-16 May 2007*. Springer-Verlag, 2007. [Online]. Available: http://sit.sit.fraunhofer.de/smv/publications/download/VoIP_Non_Repudiation.pdf

[10] C. Hett, N. Kuntze, and A. U. Schmidt, "Security and non-repudiation for voice-over-ip conversations," Poster presentation at the ISSA 2006 From Insight to Foresight Conference, Sandton, South Africa, 5th-7th July 2006, 2006. [Online]. Available: http://sit.sit.fraunhofer.de/smv/publications/download/Signed_Voice_Transactions_ISSA06_final.pdf

[11] D. Wright, *Software Life Cycle Management*. It Governance Ltd, 2011.

[12] M. Olive, R. Oishi, and S. Arentz, "Commercial aircraft information security-an overview of arinc report 811," in *25th Digital Avionics Systems Conference, 2006 IEEE/AIAA*. IEEE, 2006, pp. 1–12.

[13] R. De Cerchio and C. Riley, "Aircraft systems cyber security," in *Digital Avionics Systems Conference (DASC), 2011 IEEE/AIAA 30th*. IEEE, 2011, pp. 1C3–1.

[14] R. Robinson, M. Li, S. Lintelman, K. Sampigethaya, R. Poovendran, D. Von Oheimb, J.-U. Bußer, and J. Cuellar, "Electronic distribution of airplane software and the impact of information security on airplane safety," in *Computer Safety, Reliability, and Security*. Springer, 2007, pp. 28–39.

[15] P. Skaves, "Information for cyber security issues related to aircraft systems," in *Digital Avionics Systems Conference (DASC), 2011 IEEE/AIAA 30th*. IEEE, 2011, pp. 1C2–1.

[16] C. A. S. Authority, *CAAP 232A - 1(0) – Administration of Aircraft & Related Ground Support Network Security Programs*. Australian Government, 2013.