Securing a Dependability Improvement Mechanism for Cyber Physical Systems

https://americancse.org/events/csce2020/conferences/serp20

Gilbert Regan1,\* ,Fergal Mc Caffery1 Pangkaj Chandra Paul1 Ioannis Sorokos 2 Jan Reich 2 , Eric Armengaud 3, Marc Zeller4 ,

1 Lero @DKIT, Dundalk, Ireland

Gilbert.regan,fergal.mccaffery and pangkajchandra.paul@dkit.ie

2 Fraunhofer IESE, Kaiserslautern, Germany
jan.reich and ioannis.sorokos@iese.fraunhofer.de

3 AVL List GmbH, Austria, Turkey
eric.armengaud@avl.com

4 Siemens, Munich, Germany
marc.zeller@siemens.com

Correspondence gilbert.regan@dkit.ie

**Abstract.** The open and cooperative nature of Cyber-Physical Systems (CPS) poses a significant new challenge in assuring dependability. A European funded project named DEIS addresses this important and unsolved challenge by developing technologies that facilitate the efficient synthesis of components and systems based on their dependability information. The key innovation that is the aim of DEIS is the corresponding concept of a **D**igital **D**ependability **I**dentity (DDI). A DDI contains all the information that uniquely describes the dependability characteristics of a CPS or CPS component.

In this paper we present an overview of the DDI, and provide the protocol for ensuring the security of the DDI while it is in transit and rest. Additionally, we provide confidentiality, integrity and availability validation of the protocol.

**Keywords:** dependability, cyber physical system, evaluation, cyber security.

1. Introduction

Cyber-Physical Systems (CPS) harbor the potential for vast economic and societal impact in domains such as mobility, home automation and delivery of health. At the same time, if such systems fail they may harm people and lead to temporary collapse of important infrastructures with catastrophic results for industry and society [1]. There are two core challenges while assessing the dependability of a CPS. First, the inherent complexity of modern CPS [2] and the resulting complex market organisation requiring the tight cooperation between different teams, expertise, and institutions, while managing confidentiality issues. The second challenge is related to the increase of connectivity, e.g., through machine to machine cooperation enabled by Internet of Things, which introduces a new dynamic in system operation [2]. As a result, Cyber-Physical Systems of Systems (CPSoS) come together as temporary configurations of CPS, and which dissolve and give place to other configurations. This leads to a potentially infinite number of variants, with cooperation between systems potentially not analysed during design time.

The DEIS project [3] addresses these important and unsolved challenges by developing technologies that form a science of dependable system integration. In the core of these technologies lies the concept of a Digital Dependability Identity (DDI) of a component or system. The DDI targets (1) improving the efficiency of generating consistent dependability argumentation over the supply chain during design time, and (2) laying the foundation for runtime certification of ad-hoc networks of embedded-systems.

Contribution of this paper is to present the protocol for securing the DDI while it is in transit and at rest. The paper is organized as follow: Section 2 presents an overview of the DDI while the research methodology is presented in Section 3. Section 4 presents the protocol for securing the DDI while it is in transit, while section 5 presents the protocol for securing the DDI while it is at rest. Finally, Section 6 presents validation results while section 7 concludes this work.

1. Overview of DDI (1 page)

Assurance cases represent the backbone of modern dependability assurance processes, because they record the dependability requirements to be fulfilled by a system (of system) in an intended operational environment together with the evidences that support the requirement’s validity in the finally implemented system. All produced dependability-engineering artifacts using such evidence are motivated by an uncertainty about whether a dependability claim about the system is actually fulfilled.

Since there is an interrelation between the system, its dependability claims, and the supporting evidence artifacts that exist in the real world, we claim this should also be the case for the system’s model-based safety reflection, i.e. its DDI (see **Fig. 1**).

Certification Activities/ Standards

**Open Dependability Exchange Metamodel (ODE)**

DDI

*instance of*

TARA

interface

Failure Logic

interface

HARA

interface

Architecture

interface

**Dependability**

Assurance Case

artifacts

terminology

**Fig. 1.** The Open Dependability Exchange Metamodel (ODE)

DDIs represent an integrated set of dependability data models that may be (semi-)automatically analysed, generated or manipulated during the execution of safety engineering processes. A DDI contains information that uniquely describes all the dependability characteristics of a system required for certifying the system’s dependability. DDIs are formed as modular assurance cases and their composability allows for the (semi-)automatically synthesizing of system DDIs from the DDIs of the subcomponents. The DDI of a system contains a) claims about the dependability guarantees given by a system to other systems and derived system dependability requirements and b) supporting evidence for those claims in the form of various models and analyses. For security assurance, it contains a threat and risk analyses (TARA) which is composed of attack trees, while for safety assurance, hazard and risk analyses (HARA), architecture modeling and failure propagation modeling such as fault trees, FMEA or Markov chains are supported.

Due to the integration and standardisation of these models in the Open Dependability Exchange Metamodel (ODE), a self-contained system dependability package can support many dependability-engineering activities of the system lifecycle. A video of the DDI being employed in a truck platooning use case can be seen here [4].

1. Methodology

The data confidentiality, integrity and availability (CIA) triad is a common concept to ensure data security. The attacker can launch various attacks to compromise the CIA of data in-transit and at-rest. To mitigate these attacks and to assure the CIA of the DDI, this research designed a security protocol which is presented in **Fig 2**. The proposed security protocol consists of three key stages; 1) Identify the possible threats, 2) Identify the security controls and 3) Evaluate the security of DDI.



**Fig. 2.** DDI security protocol

Risk assessment process such as NIST 800-30 [4], CIS RAM [5] or a threat modeling technique such as STRIDE [6] can be used to identify the possible threats. As the focus of this research is on securing the DDI and not the whole application, attacks such as Denial of Service (DOS), eavesdropping and data modification are considered.

The next stage in the protocol is to identify mitigating security controls. There are several standards, guidelines and frameworks which provide the security controls to put countermeasure against various attacks. Examples include ISO/IEC 27002 [7], NIST 800-53 [8], and the NIST Cybersecurity Framework [9]. This research adopted ISO 27002 as a source for selecting security controls because it is a widely used data security standard. This standard provides a large list of security controls with very high-level implementation details. Exclusion criteria, as detailed in Figure 2, were used to select the appropriate security controls for assuring the security of the DDI in-transit and at-rest. For example, controls related to business and management operation, or to personal security were excluded. This resulted in the following four key security control categories: Access control; Cryptography; Physical and environmental security; and Communications security. With the appropriate security controls selected, the next step was to review the implementation details of each selected control. If any of the selected security controls did not have adequate implementation details, then sources external to the ISO/IEC 27002 standard was employed. For example, ISO/IEC 27002 proposes to use cryptography to assure the integrity and confidentiality of data. However, this standard does not provide enough detail for implementing cryptography in an application. Therefore, external sources such as the NIST 800-175B Cryptographic Standard [10] and the ISO/IEC 11770 Key management standard [11]were reviewed for implementation detail.

To evaluate the proposed security protocol a truck platooning use case was selected which was implemented via a simulator framework. The simulation involves two trucks, with their intercommunication implemented via the Robot Operating System (ROS) [12]. By default, messages published to a ROS topic are in plaintext. We considered the following three types of attacks: First, an attacker could attack a critical service which monitors sensor variables and force it to be deactivated. We deemed this to be an attack on the service’s availability; Second, attackers could eavesdrop on the packet traffic and capture the plaintext ROS messages which would violate the confidentiality of the platoon’s information; finally, attackers could alter the content of the exchanged ROS messages and insert incorrect or misleading information to compromise the system’s integrity.

1. Securing the DDI in Transit

DDI data can be in transit between components within a system, or between system to cloud server, or between systems.

* 1. DDI in transit between system components

Communication between components in a system can be hardwired or wireless. Whether the connection is hardwired or wireless, the following measures are required:

Communicating components of CPS can be known (i.e. pre-certified) or unknown. If an entity is known, a pre-signed key can be used to secure communication. If an entity is unknown, the widely used Elliptic-Curve Diffie–Hellman (ECDH) protocol can be used to secure communication. This protocol is standardised by NIST in SP 800-56A, and allows two parties to establish a shared secret over an insecure channel. Basic Diffie-Hellman (instead of ECDH) can be used as well, as it has lower memory and power requirements, however EDCH produces a stronger secret key as EDCH uses algebraic curves method to generate the key.

Additionally, policies for firmware upgrade and installation, and policy for port management auditing are required. These policies assist with ensuring the confidentiality and integrity of data in transit between system components.

* 1. DDI in transit from system to cloud server

For scenarios where DDI packages are transmitted between CPS and cloud services, it is recommended to use the HTTPS (Hyper Text Transfer Protocol Secure) protocol. HTTPS establishes an encrypted link using Secure Socket Layer (SSL) or Transport Layer Security (TLS). TLS is the new version of SSL. TLS establishes an encrypted link using a TLS certificate which is also known as a digital certificate. TLS can be configured to ensure the following properties:

* Private connection via symmetric cryptography
* Authentication via public key cryptography
* Data Integrity via a Message Authentication Code (MAC).
	1. DDI in transit from system to system

Individual constituent systems can be unknown, known to each other or known centrally by some management authority. Unknown parties are inherently untrusted and are potential avenues for malicious attacks on confidentiality and integrity. Therefore, where communication involves directly or indirectly untrusted parties, exchanged data must be secured at the system boundaries. To secure the transit of data between systems, the following considerations need to be taken into account:

* Are the systems involved in the exchange pre-certified or do they need to be certified on the fly? For pre-certified systems, each CPS’ key will be stored in the Key Management Service (KMS) and can be shared from there. For systems that need to be certified on the fly, each system can generate their own encryption key and share it. Additionally, if a system is known centrally, it can retrieve the key from the KMS and share from there.
* Choice of encryption key i.e. Asymmetric or Symmetric? The choice of cryptography technique (asymmetric/symmetric) depends on device resources (i.e. computational power, memory etc.) and the KMS cost. In general, symmetric encryption requires less device resources, is less costly, and requires minimal effort for key management.
* Is the communication to be one-to-one between systems, and/or is the message to be broadcast to all systems in the network?

The following section portrays how the system to system protocol can be applied to the platoon use case.

* 1. System to System Protocol Applied to Platoon Use case

Figure 2 provides the communication model for the platoon use case. Communication can be ‘bidirectional’ or ‘bidirectional with centralized broadcast’.



Figure 2: Platoon Communication Models

The options for securing both of these communication models for pre-certified and certified on-the –fly systems are now provided.

**Bidirectional with pre-certification**

With pre-certification, there are 2 encryption options available – using a symmetric or an asymmetric key. The choice of asymmetric versus symmetric encryption is decided based on computational cost and Key management cost. The choice of encryption approach ultimately depends on trade-off analysis during development.

*Asymmetric encryption* is generally more resource intensive. For asymmetric encryption, the Central Authority always maintains a private key and each system has a public key. There are two ways to generate asymmetric keys:

* RSA – for a shared public key
* X.509 – for each system having their own individual public key

*Symmetric key encryption* means each system shares their symmetric keys with the system they want to communicate with.

For the Platoon use case with *asymmetric key encryption*, RSA with shared public key option is chosen because a Certificate Authority is required for managing X.509 certificate and deploying such a service is costly and time consuming. Using *Symmetric key encryption*, the symmetric key will be generated using AES 256 which is a widely recognised standard. Each truck shares their symmetric keys with the truck they want to communicate with.

With pre-certified key sharing, all keys are generated and stored centrally in the Key Management Service (KMS). Thus, each authenticated truck can obtain both their own key, as well as their neighbor’s truck key from the KMS.

**Bidirectional with On-the-Fly Certification**

*Asymmetric key encryption* - Public Key Infrastructure (PKI) certification (e.g. via X.509) cannot be used because a Certificate Authority is needed. Deploying such a service is too costly and time consuming for an ad-hoc network of systems. For RSA certification, a public and private key pair must be generated, between two systems.

There are many ways to exchange asymmetric keys, the following are reasonable options:

* A master key or signature (manufacturer-specific) is used to encrypt the keys and share them over the communication channel.
* Keys are shared during ACK handshake with Message Authentication Code (MAC)

For the platoon use case using Asymmetric Key Encryption, RSA certification with a public and private key pair is chosen because it is cheaper and less time consuming when compared to X.509.

* When two trucks want to communicate and share keys in this platoon scenario, there are two options which they can adopt. The choice of which option is determined by the manufacturer during development.
* Option 1: Truck 1 will use a master key or signature (manufacturer-specific) to encrypt the keys and share with the follower truck over the communication channel.
* Option 2: The two trucks can share Keys during ACK handshake with Message Authentication Code (MAC).

For the platoon use case using *Symmetric Key Encryption* - each truck must be approved by a central authority (CA) server and this server will generate symmetric key using AES 256 for each truck. The CA notifies each truck of its neighbors (if any) and shares the neighbors’ keys. At this point, each truck knows the key for its preceding and following trucks, so their bidirectional communication can be secured.

**Bidirectional with broadcast message with Pre-certification**

In this model, trucks communicate with their neighbors, but the lead truck can also broadcast to each truck. For broadcast messages from the lead truck, asymmetric encryption can be used by sharing the manufacturer’s specific public key with all members in the platoon. If symmetric encryption is preferred, a known common key must be shared with all platoon members. The techniques for exchanging keys and the use of a KMS, as described above under ‘Bidirectional with pre-certification technique’ can be applied again.

**Bidirectional with broadcast message – On-the-Fly Certification**

*Asymmetric Encryption -*For one to one communication between two platoon members, the same recommendation as per ‘Bidirectional On-the –fly’ can be used i.e. using RSA to generate public-private keys between platoon member pair.

For message broadcasting, the Leader’s public key will be shared with all members in the platoon. The same recommendations for sharing keys per ‘Bidirectional On-the–fly’ still apply.

*Symmetric Encryption* - Assumption: Any truck wanting to join platoon is approved by platoon leader. The Leader will generate each truck’s symmetric key using AES 256 and exchange using Diffie Hellman technique. The recommendations for platoon member communication are the same as per ‘Bidirectional on-the-fly’.

For Broadcast message the Leader will generate a generic key and share it with each member of the platoon separately. To share generic key securely, Leader will encrypt the generic key using each individual’s key (which was generated during one to one). This encrypted generic key can now be decrypted by each individual member.

1. Securing the DDI at Rest (Gilbert)

The DDI at rest is the case where the DDI is stored statically within a CPS, for instance, in local memory or on a cloud service. When the stored data involves intellectual property concerns, or is significant for the system’s functionality, or is personal data, then it may be prudent or even mandatory to encrypt the data.

Asymmetric and symmetric encryption are two mutually exclusive options. Asymmetric keys (also known as Public keys) require high computational power for encryption and decryption but are considered very secure. Symmetric keys are comparatively cheaper, require less computational power and introduce less communication delay. For the above reasons, symmetric keys are recommended by default for DDI applications.

There are two further options to consider with symmetric keys:

* Stream ciphers encrypt and decrypt data one bit at a time which means that they are particularly well-suited to real-time hardware-based applications, such as audio and video applications. Stream ciphers are weaker and less efficient than Block ciphers when it comes to software applications and are less frequently used in that sphere. The encryption key size is often the same length in both approaches;
* For Block ciphers, strong algorithms mean that reverse-engineering the cipher, or determining which functions were performed on each block, or their order, is virtually impossible.

For symmetric cryptographic encryption, the Advanced Encryption Standard (AES) 256 is recommended**.** AES 256 is a widely recognized symmetric key and recognized by standards bodies i.e. ISO 18033-3 (Security Techniques Standard) and NIST 800-175B (Using Cryptographic standards in the Federal Government). Symmetric key block cipher algorithms include: SEED (Block), Camellia (Block), CAST-128 (Block), Blowfish (Block), AES (Block), DES (Block).

**Encryption key storage**

After the encryption keys have been generated, consideration needs to be given as to how they will be stored in the System and in the Cloud. For cloud storage, the KMS can be used. A cloud service with FIPS 140-2 (Cryptographic Modules Standards), which use hardware security modules (HSMs) to generate and protect keys should be chosen. HSMs are considered more secure than software encryption for generating encryption keys. For system storage, the key can be stored in the EPROM Erasable Programmable Read-Only Memory (EPROM.

**Cloud-service-specific security measures**

To secure data in the cloud an ‘Encryption at Rest’ feature should be enabled. This means the hard drive in the cloud is encrypted. Additionally, for port management ensure that only those ports that you require are open. Finally, every cloud has an Identity Management Service, which needs to be configured to ensure appropriate

Identity Management for Access Control (IMAC).

**DDI File Security in the Cloud**

For files stored in Cloud ‘Storage Service’ the storage should be encrypted at file level. Additionally, files should not be publicly accessible i.e. only authorized access by application.

**Application Security**

A Web Application Firewall (WAF) should be employed in the Cloud. The WAF helps protect against attacks such as DDOS, SQL injection, Path Traversal etc. The firewall will help ensure the availability of the system.

**Database Security**

Appropriate Access Control and Role Management should be employed so that only the application has access to the database. To encrypt the hard drive of the database, it should be configured with ‘Encryption at Rest’ service. Finally, all sensitive information (Personal Identifiable Information) in the database should be encrypted using field level encryption e.g. AES 256.

1. Results

To demonstrate the attack in our use case, we used the built-in ROS command-line utility ‘rostopic echo’, that directly outputted the contents of the messages exchanged by the vehicle systems. The result can be seen on the left of Figure 4. After enabling encryption and message authentication, message contents instead only include the encrypted payload and the MAC, seen on the right side of Figure 4.



Figure - Encryption and MAC Application

To address availability, we implemented a service supervisor for the sensor monitor. The attack defended against would attempt to take down the monitor service, debilitating the vehicle. Such attacks are possible in ROS via the RosPenTo [13]. After taking down the sensor monitor, the following vehicle no longer reacts to changing conditions, leading to potentially unsafe driving behavior. The service supervisor is activatable via the simulation user interface, seen in Figure 5. Once enabled, taking down the service, leads to the supervisor detecting and re-launching the monitor, thereby ensuring availability and safe driving behavior.



Figure 5 - Partial View of Simulator Interface

1. Conclusion

Securing open and adaptive CPS is paramount to maintaining their effectiveness and delivering their full potential to users and infrastructure. The DEIS project developed the concept of the DDI to support generic information exchange across CPS. In this publication, we presented our investigation into recommended security protocols that aim to provide base coverage across a diverse set of CPS application scenarios e.g. securing the DDI in transit, at rest and more. These preliminary recommendations are by no means exhaustive, and it will be useful to expand upon them in future work.

As part of DEIS, we chose a subset of the recommended protocols to implement and evaluate within a truck platooning use case. We identified attacks that covered the standard confidentiality, integrity and availability properties of the platoon CPS. We then implemented our recommended protocols successfully against the chosen attacks. The above use case should provide a reasonable basis for security analysis and protection for applications of a similar nature to the platoon system investigated. For more diverse CPS applications, the recommended protocols presented earlier can be reviewed and adjusted to secure DDI and/or similar exchanged information concepts.

Moving forward, we will be investigating in more detail means of incorporating privacy-specific threat analyses and protection methods, such as LINDDUN [14].

**Acknowledgement** This paper is supported by the European Union’s Horizon 2020 research and innovation programme under grant agreement No 732242. It is also supported in part by Science Foundation Ireland grant 13/RC/2094.

References

1. Wei,R.,Kelly,T,.Hawkins,R, and Armengaud,E. DEIS: Dependability Engineering Innovation for Cyber Physical Systems. [Lecture Notes in Computer Science](https://link.springer.com/bookseries/558), vol 10748. Springer,Cham 2018
2. Platforms 4CPS.[Online]. Available: <https://www.platforms4cps.eu/fileadmin/user_upload/Deliverable_1.2_European_ecosystem_and_market_opportunities_assessment.pdf>[Accessed: 21-Mar-2020

3. DEIS. [Online]. Available: http://deis-project.eu/. [Accessed: 21-Mar-2020].

4. DEIS Truck Platooning. Available: https://www.youtube.com/watch?v=Vdn-TCGxzgA

5. NIST Special Publication (SP) 800-30, Rev 1, Guide for Conducting Risk Assessments. Available: https://www.nist.gov/privacy-framework/nist-sp-800-30

6. Center for Internet Security. Risk assessment Method. Available: <https://learn.cisecurity.org/cis-ram> [Accessed: 21-Mar-2020].

7. Software Engineering Institute. Threat Modelling:12 Available Methods. Available: https://insights.sei.cmu.edu/sei\_blog/2018/12/threat-modeling-12-available-methods.html . [Accessed: 21-Mar-2020].

8. ISO/IEC 27002:2013 — Information technology — Security techniques — Code of practice for information security controls.

9. NIST SP 800-53 Rev. 4 Security and Privacy Controls for Federal Information Systems and Organizations. Available: <https://csrc.nist.gov/publications/detail/sp/800-53/rev-4/final>

10. NIST Cybersecurity Framework. Available: <https://www.nist.gov/cyberframework>

11. NIST SP 800-175B Guideline for Using Cryptographic Standards in the Federal Government: Cryptographic Mechanisms. Available: <https://csrc.nist.gov/publications/detail/sp/800-175b/final>

12. ISO/IEC 11770-1:2010 [ISO/IEC 11770-1:2010] Information technology — Security techniques — Key management — Part 1: Framework

13. Robot Operating System. Available: https://www.ros.org/

14. Dieber, B.,White, R,.Taurer, S,.Breiling, B,.Caiazza, G,.Christensen, H, and Cortesi, A. Robot Operating Systems (ROS). Studies in Computational Intelligence, vol 831. Springer,Cham. 2019

15. Wuyts,K,. Scandariato,R, and Joosen,W. Empirical evaluation of a privacy-focused threat modeling methodology,Journal of Systems and Software,Vol 96, Pages 122-138. 2014.