Using Android Devices as Mobile Extensible HMIs

Michiel Willocx, Jan Vossaert, Vincent Raes and Vincent Naessens
MSEC, imec-DistriNet
KU Leuven, Technology Campus Ghent
Gebroeders Desmetstraat 1, 9000 Ghent, Belgium
Email: name.lastname@cs.kuleuven.be

***Abstract*—Manufacturing processes are currently visualised and controlled using dedicated Human Machine Interface (HMI) components. These single-use devices are typically located at fixed locations in the production hall. This poses significant constraints regarding mobility and extensibility. Smartphones can provide an interesting complementary HMI solution to those standard HMI components. Smartphones provide a mobile solution which allows engineers to access production process data (e.g. access sensor values or PLC logs) from anywhere in the field. Moreover, software on smartphones can be easily extended to provide additional functionality such as access to company services, resources on the Internet or interaction with IoT devices installed in the industrial network. This paper presents the design of a mobile extensible HMI using the Android platform. The design focuses on tackling the security requirements associated with smartphone access to the industrial control system network. The design relies on the Android for Work framework contained in the Android 5+ platform. A prototype setup is realized, illustrating the feasibility of the proposed solution.**

***Index Terms*—Android; IIoT; HMI; CPS; Case study; Best practice**

I. Introduction

Currently manufacturing companies use standard Human Machine Interfaces (HMIs) from Industrial Control System (ICS) component vendors such as Siemens, Phoenix Contact and Rockwell Automation to visualise and control the production process. These devices communicate via the ICS network with the Programmable Logic Controllers (PLCs) controlling and the sensors monitoring the production process. HMI devices are single-use devices (i.e. monitor and control the production process) and are typically located at fixed locations in the production hall. They provide operators with dedicated interfaces to monitor and control the production process during its normal operation. However, smartphones can provide an interesting complementary HMI solution to those standard HMI components. They provide a mobile platform which allows accessing production process data from anywhere in the field. This, for instance, allows engineers to remotely silence alarms or access sensor values and PLC logs when working on machines located at remote locations in the field, with no direct access to HMIs. They provide a cheap off-the-shelf alternative with rich connectivity and UI capabilities. Several standard ICS component vendors such as Siemens, Phoenix Contact and Rockwell Automation have started to provide mobile applications to interact with their devices. Moreover, apart from providing access to production data, smartphones also provide a convenient platform to access company services, resources on the Internet or IoT devices installed in the industrial network. The (controlled) installation of additional (IIoT) applications enables users to quickly have access to new services provided in the context of the ICS.

Despite the many advantages, industry is reluctant to allow mobile devices access to the ICS network due to the associated security risks. Industrial components typically have a long lifespan and rarely receive software updates [1]. Hence, many components in the field have known security vulnerabilities and are sensitive to crashes when receiving unexpected types of traffic. Moreover, commonly used ICS protocols do not provide any security guarantees [1] (e.g. message authentication). Access control to the ICS network is, hence, a crucial concern. Providing an access point for mobile devices introduces additional security risks [2] that need to be managed, especially if the mobile device is also used to run non-ICS applications. Although these non-ICS applications introduce additional security risks, they can also provide opportunities to increase the efficiency of workflows by, for instance, providing access to services in the company network (e.g. to schedule repairs, manage the production schedule or access historical production data), on the Internet (e.g. communicate with external service engineers, consult ICS device documentation or access IIoT services).

The contribution of this paper is twofold. First, the design of a mobile extensible HMI using the Android platform is presented. The design focuses on tackling the security requirements associated with smartphone access to the ICS network. The design relies on the Android for Work framework contained in the Android 5+ platform. Second, a prototype setup is realized, illustrating the feasibility of the proposed solution.

The remainder of this paper is structured as follows. Section 2 points to related work. The used technologies are introduced in Section 3, followed by the design of the system in Section 4. Subsequently, a prototype setup is presented in section 5. The design and implementation is evaluated in Section 6. Finally, conclusions are drawn in Section 7.

II. Related Work

Several research initiatives focus on connecting mobile devices with components in the ICS network. Some solutions focus on monitoring the production process from remote locations using the mobile device. Ramamurthy et al. propose [3] an SMS-based system that notifies users if a certain preset condition occurs in the ICS network. The use of one-way communication, however, limits the flexibility of the solution, and prevents it from providing the degree of access targeted in this paper. Drumea presents a solution [4] that uses the USB interface of the mobile device to interact with ICS devices. A similar approach [5] uses NFC as communication technology. This decreases the security risks since the mobile devices does not have access to the ICS network itself. However, since USB is not typically used for IO with ICS components, it requires additional custom hardware. Pethig et al. present [6] a data acquisition architecture for industrial networks. The data is gathered in a central server, providing controlled access to the data by external devices. Hence, employees can access ICS data on the production floor using mobile devices. The mobile devices do not need to have direct access to the ICS network, protecting the integrity of the ICS network. It, however, does not provide the fine grained access and control that many use cases require. Some related work [7], [8], [9] focuses on providing HMI Web services in the ICS network. This removes the need for dedicated applications to interact with ICS components from mobile devices. How mobile devices can be securely integrated in the ICS network is, however, not discussed. It is, hence, complementary to the work presented in this paper.

Related work also discusses the integration of dedicated industrial IoT devices on the production floor. These devices can be used to collect additional information about the production process and equipment to provide new types of services [10], [11] such as predictive maintenance. Introducing these additional devices increases the attack surface. Hence, several papers [12] focus on discussing the associated security techniques and present techniques to protect the integrity of the software [13], [14] running on those devices and the confidentiality and integrity of the exchanged data [15]. This paper presents complementary work, focusing on securely providing access to devices and data in the industrial (IoT) network from a general purpose end-user mobile device with a more flexible and open software model. Wireless communication is often used when deploying IoT networks. Using wireless communication technologies on the production floor, however, introduces many technical and security challenges. For instance, related work [16], [17] focuses on overcoming the many technical challenges of providing reliable wireless monitoring and control systems.

Mobile platforms such as Android and iOS use several techniques to protect the integrity of the operating systems, isolate individual applications [18] and protect the data stored [19] on the device. Several papers assess [20] the security mechanisms included in these platforms or present additional mechanisms to further increase the security of the platform. For instance, Bugiel et al. [21] introduce a security framework that assigns applications to different trust levels to increase control over application interactions. Grace et al. [22] present a zero-day malware detection mechanism based on application behavior. The system presented in this paper relies on the security of the used platform.

III. Preliminaries

Android for Work (AfW)[1] is a program for supporting enterprise use of Android. It comes built-in with Android as of version 5. It provides a set of APIs that Enterprise Mobility Management (EMM) providers and enterprise application developers can use to create a secure environment for corporate applications. It supports the creation of two separate profiles, a personal profile and a work profile. Corporate data is managed in the work profile and protected from personal applications. It, hence, supports commonly used mobile device management models such as Corporate-Owned, Personally Enabled (COPE) and Bring Your Own Device (BYOD), in which the device is used both in business and personal context [23]. To enroll a device with the MDM system, the MDM application of the selected EMM provider is installed on the device and linked to the corporate account of the user. The application uses the AfW API to create the work profile and enforce the MDM policies on the device. The application contacts the EMM server to retrieve the policies to be enforced on the device. Some policies are applied system-wide (e.g. enforce full disk encryption and password policies), other policies only apply to the work profile (e.g. credentials to access work services, specify apps that can be installed in the work profile).

The Android platform provides built-in support for IPsec, but also allows the installation of VPN applications to, for instance, add support for OpenVPN. These applications establish the secure tunnel and use the `VpnService`[2] API provided by the Android platform to register their VPN service with the platform. The system requests permission of the user the first time the VPN is established. After the VPN is established, the Android platform routes the traffic of other applications via the VPN tunnel. The VPN established by applications is restricted to the container in which the application is running. For instance, if a work VPN application is used to establish a VPN to the corporate network, only the traffic from applications installed in the work profile is sent over the VPN. Traffic from personal profile applications is still sent via the regular network interface.

IV. Design

This section first presents the threat model, subsequently the requirements are discussed and finally the used approach is presented.

A. Threat Model

It is assumed that the integrity of the Android system on the mobile devices is ensured and that the MDM policies and OS-level security measures (e.g. application sandboxing) are correctly applied. The EMM platform provides a secure platform for distributing and applying the MDM policies on the mobile devices. The system relies on standardized and commonly used cryptographic protocols and security technologies. It is

[1]More information can be found on: https://developer.android.com/work/index.html.

[2]More information can be found on: https://developer.android.com/reference/android/net/VpnService.html.

assumed that these protocols and implementations provide the proposed security properties.

The attacker model considers malicious applications that could be installed on the device, both in the personal and work profile. Although, the work profile is typically constrained by the system administrator in the apps that can be installed by the user, it still provides opportunities for attackers to attack the ICS system. For instance, some applications may contain bugs that could be exploited to modify the behavior of the application or malicious content could be injected (e.g. remote code updates) via the application back-end. Moreover, some benign applications may try to send traffic to the ICS network, which could cause undesired behavior in the ICS network. A second type of attacker attempts to gain access to the ICS system by gaining access to a mobile device with the required credentials.

B. Requirements

a) Functional requirements:

- F_1 Business apps can be used to access work resources and services.
- F_2 ICS apps can be used to interact with ICS/IIoT equipment in the ICS network.
- F_3 The user is free to install personal applications and manage personal data on the device.

b) Security requirements:

- S_1 Only a limited set of predefined company-verified apps installed on the mobile device can be used to access the ICS network. This mitigates the security risks of allowing general purpose mobile devices containing a wide range of apps to access the ICS network.
- S_2 Only company-managed devices can be used to access the ICS network.
- S_3 Only authorized personnel can access the ICS network using mobile devices

C. Approach

Figure 1 gives an overview of a typical network architecture in production companies. The ICS network can be divided in several production cells. Each cell contains an HMI, monitoring and controlling the ICS process in its production cell. These cells are connected via the ICS router. The ICS router separates the office network from the ICS network via a demilitarized zone (DMZ). The DMZ contains, for instance, services such as a historian that gathers and stores production process data. The main function of the DMZ is to allow devices in the office network to view historical production data, while enforcing that no devices from the office network have access to the ICS network.

To allow authorized employees (e.g. service engineers) to access devices in the ICS network from their mobile device, a separate wireless network is installed in the production floor and connected to the ICS router. The ICS router blocks all traffic from the wireless network to the ICS network, but provides access to services and resources in the office network and on the Internet. To allow access to the ICS network from the mobile device, the ICS router runs a VPN gateway. The router is configured to allow access from the VPN gateway to the ICS network. Hence, mobile devices connected to the wireless network can access devices in the ICS network via the VPN. Mobile devices that do not have the required credentials to connect to the VPN gateway can only access services in the office network and Internet. The mobile device management features of Android are used to ensure that only a set of specified apps that are trusted by the production company can be used to access the ICS network. How this is enforced is discussed in the following subsections. The mobile device management features are further used to enforce security policies to protect the corporate data of the device (e.g. full disk encryption, secure passcode). Although these policies are crucial for the security of the corporate data, they are not specific to the presented use case and, hence not discussed in further detail.

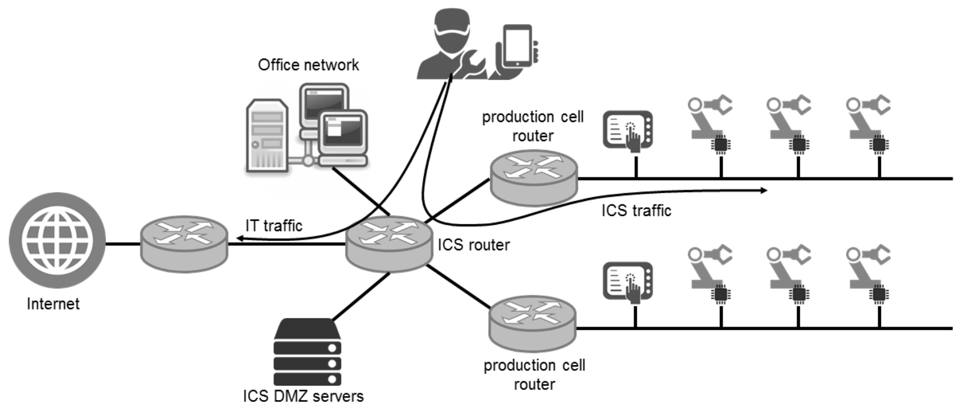
1) Restricted Application Access to the VPN: A straightforward solution to control which applications can access the ICS network via the VPN is to only allow the VPN application and the trusted ICS apps to be installed in the work profile. This, however, blocks the installation of additional apps in the work profile, effectively denying the user mobile access to, for instance, company services. It, hence, significantly decreases the potential of the mobile device, both in the context of the ICS network (e.g. scheduling repairs) and outside the context of the ICS network (e.g. accessing company email from home). A better solution is to restrict which apps in the work profile can communicate over the VPN network, and, hence, access the ICS network. For this, Android provides an API[3] that VPN applications can use to control which application's traffic is sent over VPN connections established by the VPN app. This API allows VPN applications to specify a set of *allowed applications* using the package name of the application. Only the traffic of the specified applications is redirected by the Android system to the VPN application, which, subsequently sends it over the VPN. Traffic from other applications is sent over the regular network.

2) Managed Application Configuration: For security and usability reasons, it is desirable to fully configure the VPN application via the MDM system and block user access to this information. The configuration of the VPN application consists of three main parts, namely the *VPN configuration*, *credentials* and the list of *allowed VPN apps*. The VPN configuration contains, for instance, the cryptographic parameters and the IP address of the VPN gateway. The credentials include the CA certificate and a private key and certificate to authenticate the device. The list of allowed apps consists of the package names of the applications that are allowed to access the ICS network via the VPN. In the scope of AfW, Android provides the *managed configuration* API[4] to inject the configuration information in applications via the MDM system. This API

[3]More information can be found on https://developer.android.com/reference/android/net/VpnService.Builder.html.

[4]More information can be found on: https://developer.android.com/work/managed-configurations.html.

Fig. 1. Standard production company network architecture.



is used to configure each device with its own unique settings (e.g. a unique key pair for authentication, a set of allowed VPN apps personalized for the owner of the device).

V. Prototype

This section illustrates the practical feasibility of our approach via the realisation of a prototype and discusses several implementation aspects. Figure 2 gives an overview of the prototype. It is a simplified version of the standard production company network setup, as shown in Figure 1. The Simatic S7[5] application from Siemens was used to connect with devices in the ICS network. It allows the user to connect to a PLC and, subsequently, monitor and modify variables, change the CPU operating mode and read diagnostics information. The ICS network itself consists of a SIMATIC S7-1200 PLC from Siemens. The mobile device used in the demonstrator is a Nexus 6 running Android 6. One user with access to the ICS network was added to the VPN gateway. This section further discusses several implementation aspects of the prototype. First, the use and configuration of the VPN application is discussed. Second, the configuration of the VPN gateway is presented. Finally, the use of the mobile device management software is discussed. The software developed for the prototype and the configuration of the VPN gateway can be found on https://github.com/jaert/afw-openvpn.

A. OpenVPN for Android

The prototype uses the *OpenVPN for Android*[6] application to establish a VPN tunnel with the gateway. It is a commonly used Android port of OpenVPN, and the source code is publicly available[7] on GitHub. Application support is required to use the *managed configuration* features of AfW to configure applications via the mobile device management system. Currently, few applications already support the managed configuration API. Hence, for the development of the prototype, the OpenVPN application was extended with an implementation of the *managed configuration* API. The implementation loads the values of the three configuration options (i.e. *VPN configuration*, *credentials* and the list of *allowed VPN apps*) and sets the configuration values in the application's settings. It, therefore, relies on the application's existing support for loading OpenVPN configuration files. The configuration files are separated in the general *VPN configuration* and the unique *credentials*. The two parts are Base64 encoded and loaded as string values. The added application logic loads and decodes the two configuration values and, subsequently, assembles a OpenVPN configuration file that is loaded using existing application logic to create a VPN profile. The list of allowed VPN applications is not part of the OpenVPN configuration file and, hence, handled separately. The value consists of a comma-separated list of packagenames. The extended functionality configures the created VPN profile to only allow the set of listed package names. In-app configuration of the VPN was disabled in the application. Hence, the user does not have access to the credentials used for device authentication and cannot add applications to the list of *allowed VPN apps*.

B. VPN Gateway

The ICS router was realized using a pfSense[8] distribution running on a laptop. pfSense is an open source firewall/router based on FreeBSD. It is configured to run an OpenVPN gateway, and the built-in firewall/router is configured to block traffic from the external to the ICS network and allow traffic from the VPN network to the ICS network. Access control to the VPN network is based on both user and device authentication. For device authentication a PKI infrastructure is used. A private key and certificate is issued to the gateway and the mobile devices (via the managed configuration of the VPN app) in the system. For user authentication, a list of users and associated privileges is managed by pfSense. User can authenticate using a username and password. During establishment of the VPN tunnel, the credentials of both the VPN endpoint and the mobile device are verified, as well as the username and password of the user.

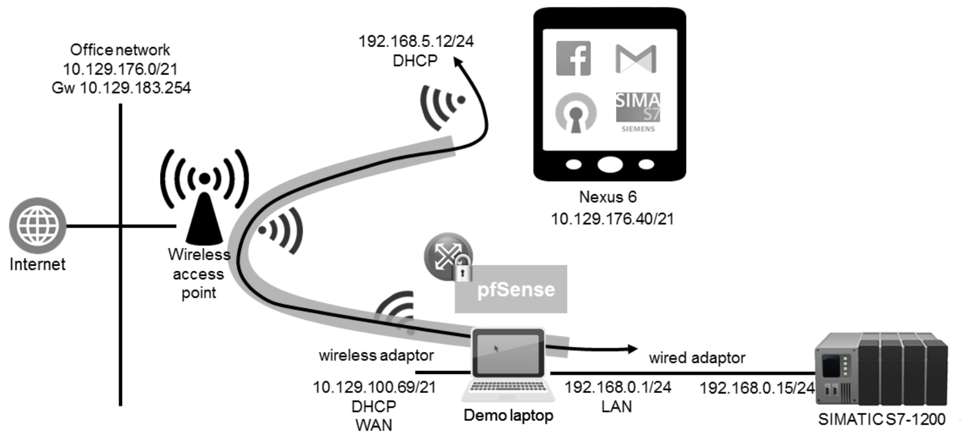
[5]The application can be found on Google Play: https://play.google.com/store/apps/details?id=com.siemens.simaticS7&hl=nl.

[6]The application can be found on Google Play: https://play.google.com/store/apps/details?id=de.blinkt.openvpn&hl=nl.

[7]The source code can be found onhttps://github.com/schwabe/ics-openvpn.

[8]More information can be found on: https://pfsense.org/.

Fig. 2. Network architecture of the proposed prototype.



C. Mobile Device Management

The MobileIron[9] cloud MDM platform is used to manage the mobile devices in the prototype. The MobileIron MDM application[10] is installed on the Nexus 6 device, and the device is enrolled with the MDM account created for the prototype. This results in the creation of a personal and work profile on the device. Via the cloud system, the list of apps that can be installed on the work profile is managed. For the prototype, this includes the modified VPN application, the Siemens Simatic S7 app, the chrome browser and a networking test applications. The VPN application is configured by setting the Base64 encoding of the required values for the three different configuration parameters (i.e. *VPN configuration*, *credentials* and the list of *allowed VPN apps*) defined in de VPN application. The MobileIron platform is sufficiently flexible to allow the distribution of unique credentials to each device and user-specific lists of applications allowed to access the VPN. The Simatic S7 is granted access to the VPN. This allows testing that the security requirements are correctly enforced in the prototype.

VI. Evaluation

A. Requirements Review

1) Functional Requirements.: Traffic of non-ICS applications is sent via the regular network interface and can, hence, access company/Internet services and resources (if reachable via the used gateway) from the mobile device (satisfying F_1 and $F2$). If the user needs access to the ICS network, he can activate the VPN via the used VPN app. When the connection is successfully established, the traffic of the set of specified ICS apps is sent over the VPN, providing access to the ICS network (satisfying F_3). In the prototype, the Simatic application can access the PLC in the ICS network if the VPN is enabled. The browser can be used to gain access to both internal and Internet Web resources.

[9]More information can be found on: https://www.mobileiron.com/.

[10]It can be found on: https://play.google.com/store/apps/details?id=com.mobileiron.anyware.android.

2) Security Requirements.: The VPN gateway providing access to the ICS network implements both device and user authentication. Since the credentials for device authentication are automatically set via MDM and are not accessible to the user, device authentication ensures that only devices that are enrolled in the MDM system can successfully establish a VPN connection (satisfying S_2). The MDM system configures the used VPN application to only allow a limited set of ICS apps to send traffic over the VPN (satisfying S_1). Since the in-app configuration of the VPN app is removed, users cannot copy the used configuration information to other devices. User authentication ensures that only authorized users can use these devices to access the ICS network. If user access needs to be revoked, their access rights can be modified on the VPN endpoint (satisfying S_3). The prototype was tested to verify that the security requirements were met. Devices without a valid private key were denied access to the VPN. The Simatic application could only access the PLC if the VPN was enabled. While the VPN was enabled, neither the networking test application in the work profile nor the Simatic application in the personal profile could access the ICS network. If the user was removed from the VPN gateway, access to the VPN was denied. A network capture taken during the testing showed that no undesired traffic was sent over the ICS network.

B. Discussion

1) Threat Model.: The assumption was made that the integrity of the operating system of the mobile devices in the system is maintained and that the OS correctly enforces the built-in security measures and policies set by the MDM system. It is, hence, important to select devices that receive frequent security updates and to ensure that they are installed on the devices. This can be enforced via the MDM system. Full disk encryption, enabled by default in recent versions of Android, and verified boot, supported by recent versions of Android, provide further protection against the undesired modification of the Android platform. If the mobile device is compromised, the credentials of the mobile device should be revoked. Logging functionality can be enabled on the VPN endpoint to track operations executed on the ICS network by each user via a mobile device.

2) Extensions.: Currently management of authorized users is done on the VPN endpoint itself. A more user friendly and flexible solution would be to use the existing corporate authentication infrastructure. This allows personnel management to centrally manage access to the ICS network and, for instance, grant technicians from external service companies or ICS device vendors short-lived access to the ICS. It can also be useful to constrain the access of authorized personnel based on their work schedules. Hence, mobile devices and credentials of authorized personnel cannot be used to gain access to the ICS network after hours. pfSense, for instance, provides authentication modules that use LDAP and Radius authentication servers.

The managed configuration API is in the proposed system only used to configure the VPN application via the MDM platform. It can, however, provide interesting opportunities to tailor the behavior of ICS and other apps installed in the work profile to the needs and requirements of the corporate system. For instance, the functionality provided by the ICS apps can be restricted to match the responsibilities of each (type of) user. Maintenance personnel can, for instance, only read the status information of ICS devices while engineering personal can also modify several parameters. This, however, requires application developers to implement the managed configuration API and to specify specific types of functionality that could be remotely enabled/disabled.

VII. Conclusion

This paper presented the design of a mobile extensible HMI using the Android platform. The design focused on tackling the security requirements associated with allowing smartphone access to the ICS network. It relies on the Android for Work framework contained in the Android 5+ platform. A prototype setup is realized, illustrating the feasibility of the proposed solution. The design strongly relies on the managed configuration features provided by the Android for Work platform. These features provide interesting opportunities to tailor the behavior of applications based on the context and privileges of the user in ICS settings and, more general, in corporate settings.

References

[1] B. Galloway and G. P. Hancke, "Introduction to industrial control networks," IEEE Communications Surveys Tutorials, vol. 15, pp. 860–880, Second 2013.

[2] S. Karnouskos, "Stuxnet worm impact on industrial cyber-physical system security," in IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society, pp. 4490–4494, Nov 2011.

[3] B. Ramamurthy, S. Bhargavi, and R. ShashiKumar, "Development of a low-cost gsm sms-based humidity remote monitoring and control system for industrial applications," International Journal of Advanced Computer Science and Applications, vol. 1, no. 4, 2010.

[4] A. Drumea, "Control of industrial systems using android-based devices," in Proceedings of the 36th International Spring Seminar on Electronics Technology, pp. 405–408, May 2013.

[5] M. Schmitt, G. Meixner, D. Gorecky, M. Seissler, and M. Loskyll, "Mobile interaction technologies in the factory of the future," IFAC Proceedings Volumes, vol. 46, no. 15, pp. 536 – 542, 2013.

[6] F. Pethig and O. Niggemann, "A process data acquisition architecture for distributed industrial networks," in Embedded World Conference 2012, Mar 2012.

[7] T. Lojka, M. Mikuf, and I. Zolotov, "Service oriented architecture for remote machine control in ics," in Applied Machine Intelligence and Informatics (SAMI), 2014 IEEE 12th International Symposium on, pp. 327–330, Jan 2014.

[8] D. Idoughi, M. Kerkar, and C. Kolski, "Towards new web services based supervisory systems in complex industrial organizations: Basic principles and case study," Comput. Ind., vol. 61, pp. 235–249, Apr. 2010.

[9] T.-H. Kim, "Scada architecture with mobile remote components," WSEAS Trans. Sys. Ctrl., vol. 5, pp. 611–622, Aug. 2010.

[10] N. Jazdi, "Cyber physical systems in the context of industry 4.0," in Automation, Quality and Testing, Robotics, 2014 IEEE International Conference on, pp. 1–4, May 2014.

[11] D. Zuehlke, "Smartfactorytowards a factory-of-things," Annual Reviews in Control, vol. 34, no. 1, pp. 129 – 138, 2010.

[12] A. R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial internet of things," in 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), pp. 1–6, June 2015.

[13] F. Brasser, B. E. Mahjoub, A. R. Sadeghi, C. Wachsmann, and P. Koeberl, "Tytan: Tiny trust anchor for tiny devices," in 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), pp. 1–6, June 2015.

[14] J. Noorman, P. Agten, W. Daniels, R. Strackx, A. Van Herrewege, C. Huygens, B. Preneel, I. Verbauwhede, and F. Piessens, "Sancus: Low-cost trustworthy extensible networked devices with a zero-software trusted computing base," in Proceedings of the 22Nd USENIX Conference on Security, SEC'13, (Berkeley, CA, USA), pp. 479–494, USENIX Association, 2013.

[15] A. Esfahani, G. Mantas, R. Matischek, F. B. Saghezchi, J. Rodriguez, A. Bicaku, S. Maksuti, M. Tauber, C. Schmittner, and J. Bastos, "A lightweight authentication mechanism for m2m communications in industrial iot environment," IEEE Internet of Things Journal, pp. 1–1, 2017.

[16] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approaches," IEEE Transactions on Industrial Electronics, vol. 56, pp. 4258–4265, Oct 2009.

[17] J. kerberg, M. Gidlund, and M. Bjrkman, "Future research challenges in wireless sensor and actuator networks targeting industrial automation," in 2011 9th IEEE International Conference on Industrial Informatics, pp. 410–415, July 2011.

[18] W. Enck, M. Ongtang, and P. McDaniel, "Understanding android security," IEEE Security Privacy, vol. 7, pp. 50–57, Jan 2009.

[19] P. Teufl, A. Fitzek, D. Hein, A. Marsalek, A. Oprisnik, and T. Zefferer, "Android encryption systems," in 2014 International Conference on Privacy and Security in Mobile Systems (PRISMS), pp. 1–8, May 2014.

[20] S. Fahl, M. Harbach, T. Muders, L. Baumgärtner, B. Freisleben, and M. Smith, "Why eve and mallory love android: An analysis of android ssl (in)security," in Proceedings of the 2012 ACM Conference on Computer and Communications Security, CCS '12, (New York, NY, USA), pp. 50–61, ACM, 2012.

[21] S. Bugiel, L. Davi, A. Dmitrienko, S. Heuser, A.-R. Sadeghi, and B. Shastry, "Practical and lightweight domain isolation on android," in Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, SPSM '11, (New York, NY, USA), pp. 51–62, ACM, 2011.

[22] M. Grace, Y. Zhou, Q. Zhang, S. Zou, and X. Jiang, "Riskranker: Scalable and accurate zero-day android malware detection," in Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services, MobiSys '12, (New York, NY, USA), pp. 281–294, ACM, 2012.

[23] M. Eslahi, M. V. Naseri, H. Hashim, N. M. Tahir, and E. H. M. Saad, "Byod: Current state and security challenges," in Computer Applications and Industrial Electronics (ISCAIE), 2014 IEEE Symposium on, pp. 189–192, April 2014.