Security considerations in modular mobile manipulation

Bernhard Dieber and Benjamin Breiling
Institute for Robotics and Mechatronics
JOANNEUM RESEARCH
Klagenfurt, Austria
Email: {firstname.lastname}@joanneum.at

***Abstract*—Mobile manipulation will play an essential role in future production's intralogistics. In addition, it can be assumed that location-independence of manipulation will greatly contribute to flexible production and higher efficiency in robot use. Modular mobile manipulators can be combined from standalone robotic components like mobile platforms and serial arms. This combination enables more flexibility since the robot can be adapted to specific use-cases by exchanging hardware. However, since those robots tend to be very complex systems, their integration into networked Industry 4.0 environments will also cause security risks. In this paper, we present a security architecture and secure interaction workflows for modular mobile manipulators that on the one hand secure the system against unauthorized manipulation and on the other hand enable the integration of mobile manipulators into larger IT infrastructures. Using the example of our inhouse-developed CHIMERA mobile manipulator, we show which architectural means can be considered in order to make a modular mobile manipulator secure. We analyze the expected attack vectors on mobile manipulators and describe their mitigation within our architecture.**

I. Introduction

Mobile manipulation [1] is a technology that now starts to penetrate manufacturing environments in course of the trend towards Industry 4.0. A mobile manipulator is the combination of one or more serial robot arms and a mobile base. It enables the robot arm to perform location-independent manipulation and thus greatly increases the working area of the robot. In order to be more flexible in production, mobile manipulation will be a key technology in intralogistics and flexible manipulation.

However, as production also becomes more networked and interconnected, factories as well as the machines in the production are increasingly becoming targets of cyber attacks. This also includes the robots employed in production that used to work in network isolation but are now becoming more and more connected. Mobile manipulators are additionally exposed due to their mobility since it enables easier physical access to the robot. Just as stationary robots and machines, also mobile manipulators need to be secured against hacking attacks. Being especially complex machines, security for modular mobile manipulators requires special consideration. Modularity enables the application-specific configuration of a robot [2] i.e. for a mobile manipulator, the exchange of the mobile base, the robot arm, sensors and other components according to application requirements. This greatly increases the flexibility and potential fields of use but at the same time makes securing the robot even harder.

In previous work, we have shown how to secure robots running ROS [3]–[5]. However, securing the middleware is only one part of a complete security strategy. Embedding it into the overall architecture and the use of the robot is at least of the same importance. In this paper, we present a security architecture and secure interaction workflows for modular mobile manipulators. We use the CHIMERA mobile manipulator (see section II-B) as an example however, our contributions can be generalized to any robot of the same type. This paper is structured as follows: we first survey the state-of-the-art in robot security and describe the CHIMERA robot in more detail in section II. We then deduct some security requirements for an architecture and workflows in section III. We present our security architecture in section IV and the associated secure interaction workflows in section V. We discuss attack vectors on mobile manipulators and how they are prevented by our approach in section VI before we conclude the paper.

II. State of the art

A. Robot security

The security in industrial systems has been a much discussed topic for years [6]–[8] also fuelled by more and more frequent security breaches [9]–[11]. The security of robots however, has long been in a niche. This is mostly caused by the fact that traditional industrial robots were not networked and only communicated with automation equipment in their direct vicinity over dedicated channels.

With the advent of more intelligent robots though, where the computing power stems from general-purpose devices and where the application environments are highly networked, security becomes a major topic of interest. As the Robot Operating System (ROS) becomes more popular also outside of research and its industrial application starts to gain speed, also the security of ROS has attracted researchers' interest. In [12], the authors have shown that ROS suffers from severe security deficits and that it can be manipulated quite easily. As shown in [13], there are already freely accessible ROS instances in the internet that can be potential targets for attackers.

There have been several approaches presented that aim at improving security in ROS. SROS is an attempt to secure ROS at the graph level and on the data communication level [14], [15]. An application-level approach has been presented in [16] where a dedicated node subscribes, encrypts and re-publishes ROS messages. At a closer look however, this approach is not effective since the plain-text ROS topic is still available for subscription.

In our previous work, we have presented an application-level approach where a dedicated authorization server is used to ensure that only valid nodes participate in the ROS network. Topic-specific encryption keys are used to ensure data confidentiality [3]. We have further extended the ROS core with authentication, authorization and encryption functions that are transparent to the ROS nodes and thus do not require nodes to be changed [4]. We have further presented initial workflows for the interaction with a secured robot that we also extend in the present paper [5]. In [17], the various approaches on ROS security are compared and evaluated.

The successor of ROS1 is currently being developed based on DDS [18] (the OMG specification for a data distribution service). Also there, security is important. In ROS2, security has received more focus so far and the proven DDS security mechanisms are used [19]. Those security enhancements are made available to ROS2 via the SROS2 project[1] and there is ongoing work on security mechanisms for ROS2 [20]. An initial performance evaluation of security in ROS2 has been presented in [21].

Independent of ROS, recently, Vilches et al. have progressed towards quantifying (in)security of robots and have presented a framework for security assessment [22], [23].

B. The CHIMERA mobile manipulator

The mobile manipulator CHIMERA is a mature research platform for mobile manipulation developed at the Institute for Robotics and Mechatronics at JOANNEUM RESEARCH. It combines a robotic arm (in the default configuration a Universal Robots UR10) and a mobile base (MiR 100 as default). Both are integrated using their native mechanisms i.e., the UR is accessed using a network port and by sending URScript commands and the MiR is accessed via ROS. The default configuration of this robot is shown in figure 1.

In order to enable a wide variety of applications for the mobile manipulator, CHIMERA's hardware configuration can be changed to reflect the requirements of a specific use-case. Fundamentally, CHIMERA is the contents of the black box between the serial manipulator and the mobile base (as shown in figure 1), the arm and platform themselves are merely exchangeable subsystems.

On the software side, a multi-layer architecture is employed to abstract the robotic hardware. As shown in figure 2, there are hardware driver modules in the lowest layer taking care about the access to MiR100 and UR10 (or another mobile base and arm) as well as camera data processing, gripping and any other hardware devices. Above that, a layer for workflow management and state propagation provides a unified interface to the application layer hiding the specific hardware used. On top, client libraries, a planner or user interaction components use this abstracted interface to interact with the CHIMERA robot.

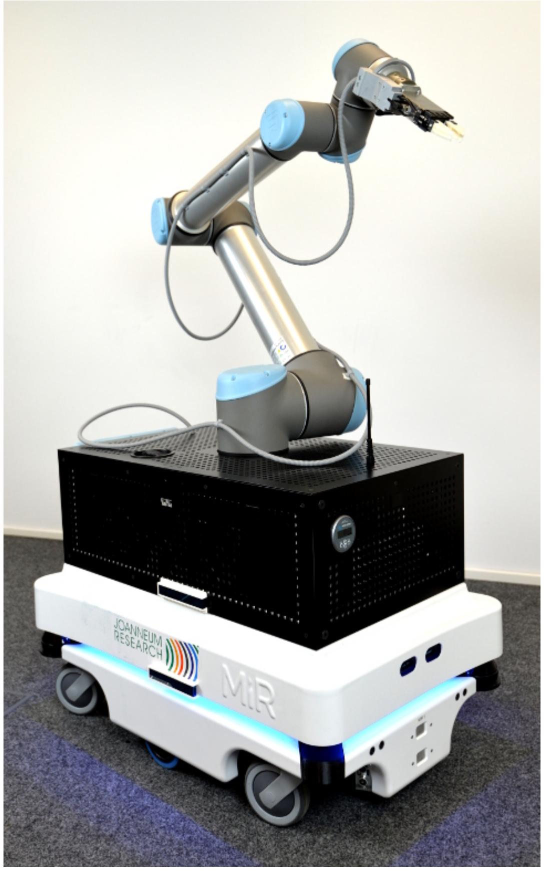

Fig. 1. The CHIMERA mobile manipulator in its default configuration with UR10 arm and MiR100 mobile base.

III. Security requirements for industrial mobile manipulators

In this section, we present some general security requirements for mobile manipulators in industrial use. We draw the requirements from two main attack vectors: i) Physical access-based attacks and ii) attacks via a network.

Table I shows the overall requirements we have identified. First it is important, that no malicious physical access is possible to the robot (Req.1). Issues like unprotected USB ports or network interfaces must be avoided. Second, the mobile manipulator must also be able to interact with external systems (Req.2). This is one important part of the robot's operation since it needs to request new tasks from an MES as well as interact with machines and users. The interaction with users and operators must also be secured to prevent any manipulation attempts (Req.3). This includes the initial commissioning of the robot, the reprogramming by shopfloor

[1] https://github.com/ros2/sros2

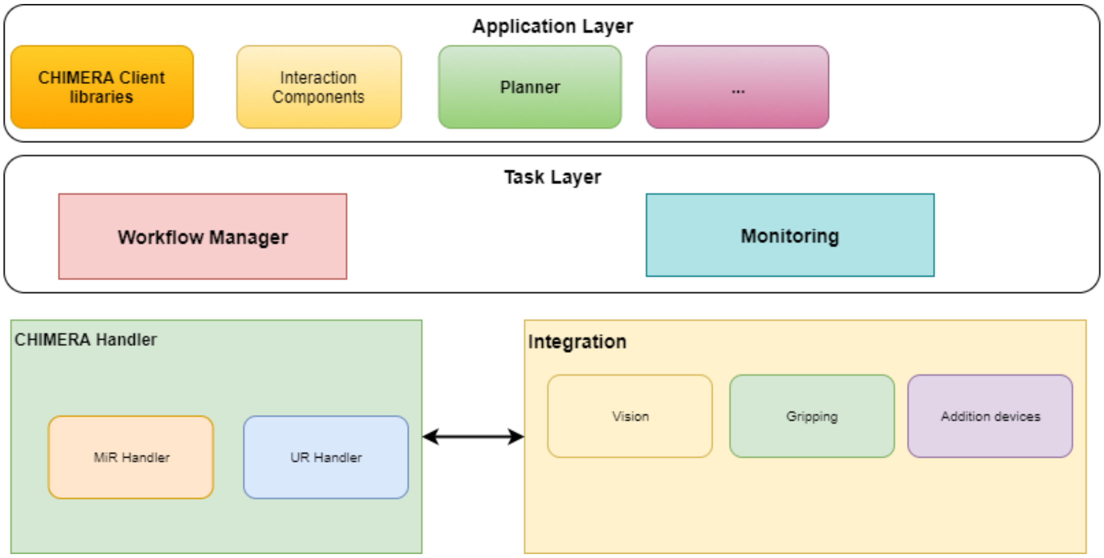

Fig. 2. The multi-layer software architecture in CHIMERA.

workers and also the maintenance (in case of reconfigurations or software updates).

No.	Name	Rationale
1	Restriction of physical access	The less physical interfaces there are, the less attack points exist
2	Interaction with external systems	The robot must be able to interact with external systems like remote control, task servers, manufacturing execution systems or similar
3	Secure user interaction	A user should easily but also securely interact with the mobile manipulator in scenarios for commissioning, maintenance or reprogramming
4	Limitation of security risk propagation	If one subsystem is penetrated, this should not infringe the other subsystems
5	Interaction of subsystems	A flawless interaction of subsystems must be guaranteed
6	Scalability	The security architecture should not restrict the scalability of software or the extension of the hardware infrastructure

TABLE I
SECURITY REQUIREMENTS FOR MOBILE MANIPULATORS.

A. *Requirements for modularity*

In case the mobile manipulator supports the modular exchange of single parts like CHIMERA does, this modularity also introduces additional security risks. A different hardware configuration also always introduces a different software setup with potentially unknown security holes. In case a subsystem is compromised despite all precautions, an ”infection” to other subsystem should be prevented (Req.4). Challenging that, it must still be possible for the subsystems to interact with each other (Req.5). Finally, the security architecture should not (or as little as possible) restrict the scalability in terms of hardware extension and software configuration (Req.6).

IV. THE CHIMERA SECURITY ARCHITECTURE

Our security architecture for the CHIMERA robot is based on subsystem isolation with only defined channels between the subsystems. In addition, we separate the software stack onto different physical devices making it harder for attackers to penetrate other subsystems. On the devices themselves, we use AppArmor[2] to define fine-grained permissions for users and processes. AppArmor is a mandatory access control system, that allows the isolation of individual applications in terms of access to other resources (like files and devices). As explained below, we use it to selectively give access to CHIMERA internals to system integrators but prevent the modification of critical system parts. In addition, this is an additional mean of isolation in line with requirement 4 from section III, table I.

Since system integrators use the mobile manipulator as a robot in various contexts, they must be enabled to add their hardware and software to the robot while still maintaining a high level of security. Thus, a system integrator has sufficient access to the robot's internals to change hardware and add drivers but e.g., not to exchange the software stack itself. Figure 3 gives an overview on how the software and hardware parts are allocated within the security architecture. We go into details on each part in the following sections. The blue solid line around the CHIMERA computing unit shows all software parts that run on this unit. Accordingly, the red dashed line around the integration computing unit shows the software parts assigned to that unit.

A. *Physical separation*

Internally, CHIMERA consists of multiple computing units with different tasks. By this, we achieve a physical separation of the CHIMERA firmware and the system integration domain.

[2] https://www.apparmor.net

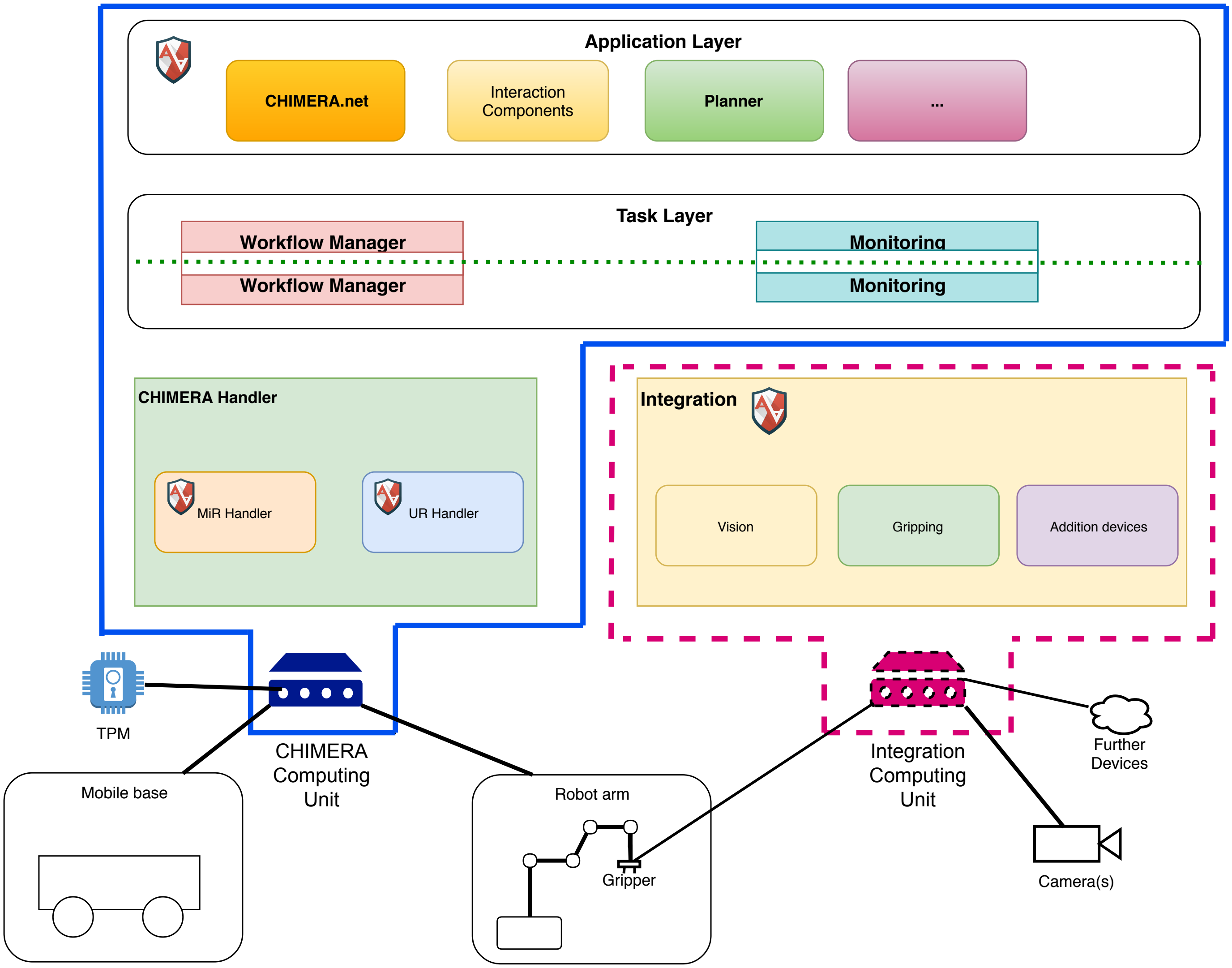

Fig. 3. The allocation of the software architecture and hardware to the computing devices

Specifically, we separate computing into a CHIMERA computing unit and an integration computing unit. The integration computing unit hosts all externally connected hardware that is not part of the core components. On this host, also additional devices can be connected. By this, a system integrator can integrate use-case specific hardware without the risk of compromising main system components. Also the gripper of the robot arm is controlled from this device, since it is typically application-specific. Network-wise, the integration computing unit is only connected to the CHIMERA computing unit and has no outside network access.

On the CHIMERA computing unit runs our main software stack along with the driver modules for mobile base and arm. The system integrator has access to this system part only to deploy software to the application layer and to deploy add-ons to the task layer. In addition, this device has an integrated trusted platform module (TPM), which is a secure cryptographic storage. On initial shipment, it contains digital certificates of authenticity for the robot itself provided by the OEM. During operation, it contains organization-specific certificates for authentication towards users and external systems. The handling of certificates is explained in more details in section V.

B. Communication channels

Between all layers, there are well-defined and security-enhanced communication channels. Outside of those channels, we prevent any access to individual system components. The application layer communicates with the task layer via a TLS-enabled [24] JSON-RPC[3] interface. Via this interface, control of all robot components is possible like joint and cartesian movement of the arm. The JSON-RPC interface uses API-Tokens for authorization of individual components. Thus, only legitimate clients get access to the robot. To add new components to the system (e.g., access extra hardware on the integration computing unit via the task layer), the system

[3] https://www.jsonrpc.org/

integrator performs the connection and driver installation on the integration computing unit, then adds code to the workflow manager that dynamically extends the JSON-RPC interface and makes it accessible to the application layer. This code however, has to be digitally signed by the system integrator and only code with a valid signature is loaded into the runtime.

The task layer (as indicated in figure 3) is split into two parts. On the computing unit, this corresponds to two separate processes that run in separated application domains. They communicate via unix sockets. This enables us to control the information flow across the task layer and to define individual security profiles for each of these processes, as they are part in different physical networks inside the CHIMERA as well. This particularly means that the hardware layer and the related network stays secured, even if the process running in the upper part of the task layer is compromised by an attacker (see Req.4 in Table I).

The communication within the hardware and integration layer is realized by a ROS network where the MIR 100 runs the ROS master. To control a system integrator's access to the MiR100 and the UR10 we provide a well defined non-ROS interface to the hardware driver modules running on the CHIMERA computing unit and deny direct network access to UR10 and MiR100. Doing so, we provide protection for our own robotic hardware from improper use committed by subsequently integrated software modules. There is a special signalling interface between the integration and the chimera hardware sections that allows the access to the UR10 signals needed to control a gripper. This is however, limited to doing just that and is disabled in cases where it is not needed (e.g., for grippers that are connected directly to the integration computing unit). The communication from and to the task layer is also running over ROS communication channels. This includes the execution of robotic tasks initiated by the workflow manager and the monitoring of the current state of the CHIMERA. With this isolation approach, we can harness the power of ROS (despite it being an insecure framework) in one part of our robot without compromising the security of the overall system.

C. AppArmor domains

We use AppArmor on multiple places in our architecture. Here, we describe the usage wherever an external user (in our case the system integrator) is concerned with that.

On the integration computing unit, an AppArmor profile ensures, that on this unit, only predefined access can be performed. The configuration can also be dependent on the use-case e.g., for a use-case where no USB device is used, the access to this can be prevented by AppArmor. Each process is run only with very fine-grained permissions to reduce the number of possible exploits.

The application layer is also secured using AppArmor. Here, the use-case specific applications of the system integrator as well as interaction components and potentially outside communication channels are realized. Thus, it is especially required to be secured since this is potentially the most exposed part of our architecture. Like the upper part of the task layer before, we additionally restrict the network access of application layer components, which means that they cannot directly connect to the hardware and integration network. Furthermore we consider to use a Web Application Firewall to protect processes opening a network connection to the environment.

Within the CHIMERA handler, we use AppArmor to secure the specific drivers of the robot arm and the mobile base. In addition, the CHIMERA handler is isolated from a network point of view. In the default configuration, we use a MiR robot as mobile base and the MiR handler uses this to control the mobile base. This robot is based on the Robot Operating System (ROS), which—despite its advantages as a prototyping platform—is a highly vulnerable system [5]. In order to incorporate this into a secure robot, the insecure part is isolated within this architecture and access to it is controlled using AppArmor.

V. Security-enabled interaction with mobile manipulators

A. Commissioning

Commissioning of a robot is the initial act of bringing it to operation within a specific environment. In the typical case, the robot is shipped by the OEM and taken over by a system integrator or an end user company. In our workflows for this operation, we want to ensure authenticity of the robot itself i.e., that the customer receives authentic hardware. In addition, it should be made sure, that no access to the robot is possible after commissioning, i.e., also not for the OEM (except for maintenance by the OEM, which occurs with the end user's consent and certificates). Thus, the original manufacturer certificates are replaced by institutional certificates of the system integrator or end user. Figure 4 shows an overview of the commissioning workflow. The engineer, who performs the commissioning, first connects to the administration interface. The OEM has shipped a smartcard along with the robot that is used for initial authentication, the engineer has the OEM's certificate that can be used to validate the certificate stored on the robot. If the certificate is not valid, the engineer can be sure that forged or modified hardware has been delivered.

B. Maintenance and reconfiguration

Since mobile manipulators are enablers of flexible production, it is to be expected that they will be reconfigured, maintained and updated in regular intervals. This should be done in a secure manner but must also be made easy for the personnel involved. The workflow is shown in figure 5. The robot is first put out of operation (e.g., by an interface button or any remote means). Then, the operator has to authenticate towards the robot in order to access the reconfiguration interface. After successful authentication and authorization, the corresponding maintenance or reconfiguration can be made. The authorization system defines fine-grained permissions where different roles will have different access levels i.e., not everyone is allowed to update software or change critical parameters. As an example,

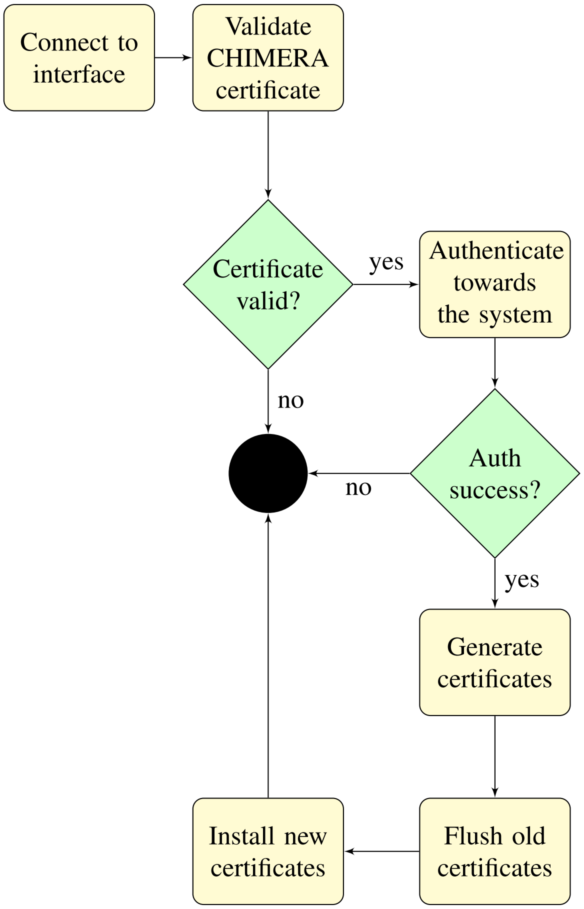

Fig. 4. Commissioning workflow.

a shopfloor worker may have access to the configuration or order of tasks of the robot while an operator may be able to change, add or remove tasks and employees of the system integrator may be able to perform software updates.

C. Decommissioning

If a robot is finally put out of operation, a secure workflow for decommissioning is required. This is required to make sure that other organizations can reuse the robot (in case it is resold) or that no sensitive material of the original organization is left on the robot. The workflow for this is similar to the commissioning workflow. However, instead of generating and uploading certificates, all existing key material is deleted from the robot along with any programs and workflows. Just like the OEM cerficates that the robot was shipped with (and that were deleted in the commissioning workflow), temporary certificates for the new owner to verify hardware integrity are generated and provided on the robot. Also, a new smart-card for initial authentication towards the system is shipped to the new owner.

D. System integrator workflow

The system integrator has to follow specific workflows when designing a use-case application. To make sure, that a high security level is maintained, sensitive areas of the robot have to be protected. After commissioning the robot, the robot has the system integrator's organizational certificates. Using those, the system integrator can access the integration computing unit and connect use-case specific hardware and add all required software components to drive this hardware. Afterwards, the integration into the task layer has to be performed. This is merely interfacing work where the system integrator exposes access to the new hardware's functions within the JSON-RPC interface using digitally signed code modules. Those code modules are deployed to the CHIMERA computing unit within a specific AppArmor domain. No access outside this is granted. Finally, the system integrator develops the use-case application on the application level. This includes user interfaces and the definition of workflows and programs that use the task layer to instruct the robot. This software is again digitally signed and deployed in a dedicated AppArmor domain. Any access to outside systems via networks is required to be configured in the AppArmor profile.

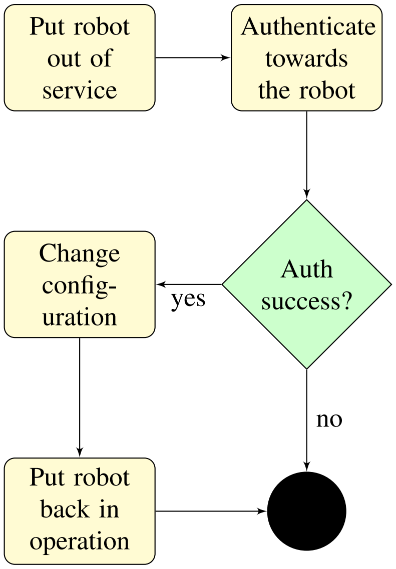

Fig. 5. Workflow for maintenance.

VI. Attack vectors and mitigation

In this section, we describe potential attack vectors on a modular mobile manipulator and how this is handled by our security architecture.

A. Physical access

The most direct way to infiltrate a robot is by having physical access to it. Typically, the first line of defence against that are organizational means i.e., no outside persons should have unsupervised access to a factory shopfloor where a robot operates. However, smaller companies often have no strict safety policy in this sense and in addition, mobile manipulation will eventually be also employed outside of industry. Thus, a mobile manipulator should be secured against physical access.

First and foremost, this requires securing the user interfaces. It should not be allowed for just anybody to read or modify data on the robot. Thus, our security architecture and workflows require users to authenticate and be authorized for every action they perform. Second, no physical interfaces should be exposed if they are not required for the current application. Especially USB ports have often served as gateway for attackers but also accessible network interfaces pose a high risk. Our AppArmor profiles can be used to restrict access to hardware interfaces. In addition, we suggest to physically obstruct unused interfaces and to disable them in the operating system configuration.

B. Remote Access via external network connections

External network connections must be secured in order to prevent a remote attack on the robot. A wide variety of scenarios with external connections is possible ranging from a smartphone-based control via the integration to manufacturing execution systems to cloud-based task scheduling. We assume however, that those specific connections are part of the application-layer modules developed by system integrators. While it stands to hope that the system integrator takes proper precautions to secure the connections, also the application itself will be restricted using AppArmor profiles thus confining any potential security breaches to this application domain.

C. Access via hardware module

A very high impact attack would be the takeover (by physical or remote means) of a hardware module like the mobile platform. As already mentioned, the MiR100 is in its default configuration a very insecure robot due to often unchanged wifi passwords and the use of the Robot Operating System. For other modern robots however, similar risks exist (e.g., the external network access to a Universal Robot is not secured by any authentication or authorization mechanism). Thus, it must be made sure that the robot component is secured by other means. For the MiR platform, we disable the MiR wifi and isolate both robots within our internal network to prevent unauthorized access to the ROS interfaces. Again, to contain potential breaches, the hardware drivers are secured by AppArmor profiles. Similar considerations and measures are taken for the hardware connected to the integration computing unit.

VII. Conclusion and future work

In this paper, we have described a security architecture for mobile manipulators in industrial use. We have shown, that modularity and security can both be supported on this kind of robot. We have further presented workflows to interact with a secured robot, that make interaction secure for both, the robot as well as the user.

We have recently presented a concept for secure state recording on intelligent robots [25]. This blackbox will be integrated into the CHIMERA robot in our future work. Here, also the integration into the security architecture and also the secure interaction workflows will be done. Further, we will validate our architecture using a team red approach where an ethical hacker without prior knowledge on the system tries to penetrate it. Any detected vulnerabilities will be incorporated into an updated architecture.

In future, also the concept for volatile memory forensics that we presented in [26] will be incorporated. Here, we continuously scan snapshots of the controller's memory for patterns of attacks in order to detect e.g., intrusions into a runnning ROS system.

In future, ROS2 will enable the robotics community to build secure robotic applications with nearly the same level of flexibility as in ROS1. In our architecture, ROS2 could provide some of the secure communication channels. However, as long as components like MiR are based on ROS1, measures for strong isolation are still mandatory. In addition, the isolation is part of the protection against zero-day issues in case vulnerabilities are present also in ROS2 / DDS.

Acknowledgment

This work has been supported by the Austrian Research Promotion Agency in the program "Production of the Future" funded project MMAssist_II (FFG No.: 858623), in the program "ICT of the Future" (grant no. 861264) and the Austrian Ministry for Transport, Innovation and Technology (bmvit).

References

[1] O. Khatib, "Mobile manipulation: The robotic assistant," *Robotics and Autonomous Systems*, vol. 26, no. 2, pp. 175 – 183, 1999, field and Service Robotics. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889098000670

[2] V. Mayoral, A. Hernndez, R. Kojcev, I. Muguruza, I. Zamalloa, A. Bilbao, and L. Usategi, "The shift in the robotics paradigm the hardware robot operating system (h-ros); an infrastructure to create interoperable robot components," in *2017 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, July 2017, pp. 229–236.

[3] B. Dieber, S. Kacianka, S. Rass, and P. Schartner, "Application-level security for ROS-based applications," in *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016)*, 2016.

[4] B. Breiling, B. Dieber, and P. Schartner, "Secure communication for the robot operating system," in *Proceedings of the 11th IEEE International Systems Conference*, 2017, pp. 360–365.

[5] B. Dieber, B. Breiling, S. Taurer, S. Kacianka, S. Rass, and P. Schartner, "Security for the robot operating system," *Robotics and Autonomous Systems*, vol. 98, pp. 192–203, 2017.

[6] E. Byres, P. E. Dr, and D. Hoffman, "The myths and facts behind cyber security risks for industrial control systems," in *In Proc. of VDE Kongress*, 2004.

[7] M. Cheminod, L. Durante, and A. Valenzano, "Review of security issues in industrial networks," *Industrial Informatics, IEEE Transactions on*, vol. 9, no. 1, pp. 277–293, Feb 2013.

[8] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, "Guide to industrial control systems (ics) security," National Institute of Standards and Technology, Tech. Rep., 2015, NIST Special Publication 800-82, Revision 2.

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

[10] N. Nelson, "The impact of dragonfly malware on industrial control systems," SANS Institute, Tech. Rep., 2016.

[11] P. Fairley, "Cybersecurity at u.s. utilities due for an upgrade: Tech to detect intrusions into industrial control systems will be mandatory [news]," *IEEE Spectrum*, vol. 53, no. 5, pp. 11–13, May 2016.

[12] J. McClean, C. Stull, C. Farrar, and D. Mascareas, “A preliminary cyber-physical security assessment of the robot operating system (ros),” in *Proc. SPIE*, vol. 8741, 2013, pp. 874 110–874 110–8. [Online]. Available: http://dx.doi.org/10.1117/12.2016189

[13] N. DeMarinis, S. Tellex, V. Kemerlis, G. Konidaris, and R. Fonseca, “Scanning the internet for ros: A view of security in robotics research,” *arXiv preprint arXiv:1808.03322*, 2018.

[14] R. White, H. Christensen, and M. Quigley, “Sros: Securing ros over the wire, in the graph, and through the kernel,” in *Proceedings of the IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS).*, 2016.

[15] R. White, G. Caiazza, H. Christensen, and A. Cortesi, “Sros1: Using and developing secure ros1 systems,” in *Robot Operating System (ROS): The Complete Reference (Volume 3)*, A. Koubaa, Ed. Springer International Publishing, 2019, pp. 373–405.

[16] F. J. Rodrguez-Lera, V. Matelln-Olivera, J. Balsa-Comern, . M. Guerrero-Higueras, and C. Fernndez-Llamas, “Message encryption in robot operating system: Collateral effects of hardening mobile robots,” *Frontiers in ICT*, vol. 5, p. 2, 2018. [Online]. Available: https://www.frontiersin.org/article/10.3389/fict.2018.00002

[17] D. Portugal, M. A. Santos, S. Pereira, and M. S. Couceiro, “On the security of robotic applications using ros,” in *Artificial Intelligence Safety and Security*. Chapman and Hall/CRC, 2018, pp. 273–289.

[18] OMG, *Data Distribution Service (DDS), Version 1.4*, Object Management Group Std., Rev. 1.4, March 2015. [Online]. Available: https://www.omg.org/spec/DDS/1.4

[19] ——, *Data Distribution Service (DDS) Security Specification, Version 1.1*, Object Management Group Std., Rev. 1.1, July 2018. [Online]. Available: https://www.omg.org/spec/DDS-SECURITY/1.1

[20] R. White, H. I. Christensen, G. Caiazza, and A. Cortesi, “Procedurally provisioned access control for robotic systems,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 1–9.

[21] J. Kim, J. M. Smereka, C. Cheung, S. Nepal, and M. Grobler, “Security and performance considerations in ros 2: A balancing act,” *arXiv preprint arXiv:1809.09566v1*, 2018.

[22] V. M. Vilches, L. A. Kirschgens, A. B. Calvo, A. H. Cordero, R. I. Pisón, D. M. Vilches, A. M. Rosas, G. O. Mendia, L. U. S. Juan, I. Z. Ugarte, *et al.*, “Introducing the robot security framework (rsf), a standardized methodology to perform security assessments in robotics,” *arXiv preprint arXiv:1806.04042*, 2018.

[23] V. M. Vilches, E. Gil-Uriarte, I. Z. Ugarte, G. O. Mendia, R. I. Pisón, L. A. Kirschgens, A. B. Calvo, A. H. Cordero, L. Apa, and C. Cerrudo, “Towards an open standard for assessing the severity of robot security vulnerabilities, the robot vulnerability scoring system (rvss),” *arXiv preprint arXiv:1807.10357*, 2018.

[24] E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.3,” Internet Requests for Comments, RFC Editor, RFC 8446, August 2018. [Online]. Available: https://www.rfc-editor.org/info/rfc8446

[25] S. Taurer, B. Dieber, and P. Schartner, “Secure data recording and bio-inspired functional integrity for intelligent robots,” in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018)*, 2018.

[26] V. M. Vilches, L. A. Kirschgens, E. Gil-Uriarte, A. Hernández, and B. Dieber, “Volatile memory forensics for the robot operating system,” *arXiv preprint arXiv:1812.09492*, 2018.