Who Controls Your Robot? An Evaluation of ROS Security Mechanisms

Niklas Goerke, David Timmermann, Ingmar Baumgart
FZI Research Center for Information Technology
Karlsruhe, Germany
{goerke, timmermann, baumgart}@fzi.de

***Abstract*—The Robot Operation System (ROS) is widely used in academia as well as the industry to build custom robot applications. Successful cyberattacks on robots can result in a loss of control for the legitimate operator and thus have a severe impact on safety if the robot is moving uncontrollably. A high level of security thus needs to be mandatory. Neither ROS 1 nor 2 in their default configuration provide protection against network based attackers. Multiple protection mechanisms have been proposed that can be used to overcome this. Unfortunately, it is unclear how effective and usable each of them are. We provide a structured analysis of the requirements these protection mechanisms need to fulfill by identifying realistic, network based attacker models and using those to derive relevant security requirements and other evaluation criteria. Based on these criteria, we analyze the protection mechanisms available and compare them to each other. We find that none of the existing protection mechanisms fulfill all of the security requirements. For both ROS 1 and 2, we discuss which protection mechanism are most relevant and give hints on how to decide on one. We hope that the requirements we identify simplify the development or enhancement of protection mechanisms that cover all aspects of ROS and that our comparison helps robot operators to choose an adequate protection mechanism for their use case.**

***Index Terms*—ROS, ROS 2, security, SROS, SROS 2, DDS**

I. Introduction

The ongoing fourth industrial revolution has brought the demand for a high level of flexibility and thus a high demand for data exchange. In industrial applications, many systems that previously operated in isolation are now connected to the Internet to be able to comply with this demand. This drastically increases their attack surface as network-based attacker can now interact with the system and either attack it from the outside or try to manipulate data that is being transmitted in between elements of the system.

The Robot Operation System (ROS) [1], first presented in 2009, is a middleware widely used for robotic applications. It has gained great interest in the field of robotics research from the very beginning. In the last years, a growing interest can also be observed in the industry. ROS provides an abstraction layer for the interaction with hardware sensors and actuators and thus drastically simplifies the development of robots and similar applications.

The use of ROS to control actuators in the physical world makes security considerations especially relevant. A loss of functionality or maliciously injected commands can have a severe impact and make the robot inflict damage on itself, the physical environment or even injure humans.

A. ROS 1

ROS 1 uses its own middleware implementation, which is not configurable. It relies on a single coordination node called "master" that controls a ROS setup and is responsible for providing parameters to the entire ROS network. Multiple ROS nodes need to be configured to perform various tasks and interact with each other using a publish-subscribe architecture. The communication in a ROS based system is implemented on top of generic networking protocols such as TCP and UDP. ROS 1 has limits of applicability, especially in the area of real-time computing and it does not provide any protection against network based attacks (see section VII-A).

The last official release for ROS 1, published in May 2020, will be maintained until May 2025. Therefore, it is necessary for users to switch from ROS to ROS 2 in the medium term.

B. ROS 2

ROS 2 is the successor of ROS and has been in development for several years now. It keeps the basic idea and many concepts of ROS (such as the publish-subscribe architecture), but also makes some major adjustments to core mechanics to meet the changing requirements. These major adjustments make ROS 2 incompatible to ROS 1. The first fundamental difference is the abandonment of the central ROS master component. Instead nodes use an auto discover feature to find each other, which makes it easier to implement a modular distributed system.

The second fundamental difference is the switch to the Data Distribution Service (DDS) [2] standard as middleware. This allows ROS 2 to be specifically configured to match the target setup via various Quality of Service (QoS) parameters. It is possible to choose between different concrete implementations of DDS from different vendors. The DDS standard provides optional security extensions that are disabled in ROS 2 by default.

C. Key Exchange Problem

Cryptographic mechanisms are used in many contexts to provide security guarantees for network communication. To do so, the sending and receiving party must have access to each other's public key or a shared secret. The exchange of these public keys or the shared secret cannot be performed via an unsecured channel such as a regular network connection where an attacker might be present. Instead, a separate trusted channel is needed, which usually requires manual interaction. To reduce the amount of key exchanges from $n!$ (where n is the amount of nodes) to n, techniques such as Public Key Infrastructures (PKIs) can be used. If a PKI or a similar technique is used, each node only needs to exchange a secret with one central trustworthy instance, nevertheless needing a separate trusted channel.

II. Our Contribution

In this work, we use an exemplary ROS networking setup to identify realistic attackers that pose a threat to ROS based robotic systems. Based on these attackers we identify relevant security objectives and two additional requirements for network-based protection mechanisms. These requirements should be used as a basis for developing protection mechanisms.

We compare ten existing protection mechanisms based on these requirements and find flaws in all of them. Finally, we discuss the results of our analysis for both ROS 1 and ROS 2, providing guidelines to help operators choose the most relevant protection mechanism for their scenario.

III. Related Work

In 2017 and 2018, DeMarinis et al [3] scanned the whole ipv4 accessible Internet for ROS based systems, finding over 140 instances. The authors were able to connect to some of the exposed instances, read sensor data and actuate a physical robot. This research shows the relevance of security measures to protect from attackers.

Multiple studies [4]–[6] have analyzed the impact of different protection mechanisms on performance. They found that some of the protection mechanisms have a significant impact on different performance measures such as CPU time or throughput. However, it is unclear which of the results can be generalized as the experiments have been conducted in testing scenarios that may not be comparable to productive ROS setups.

Clark et al [7] described and categorized attack scenarios on different kinds of robots. Additionally they described the possible economic and human safety impact of cyber-attacks on robots.

Portugal et al [5] provided an overview of security concerns in ROS and described some protection mechanisms. Their description of the existing problems identified valid points; it is not based on a structured analysis, though.

DiLuoffo et al [8], [9] analyzed the security of ROS 2 with enabled DDS security. In their analysis, they identified a few minor issues concerning the selection of cryptographic primitives and found that ROS 2 is vulnerable to an attacker that is able to access or manipulate data on the device running the ROS 2 node.

Dieber et al [10] described in detail how some attacks on ROS based systems can be performed on a technical level. The attacks regarded by the authors do not cover the whole range of what a network-based attacker as modeled in section V is able to perform, though. Their proposal to mitigate the

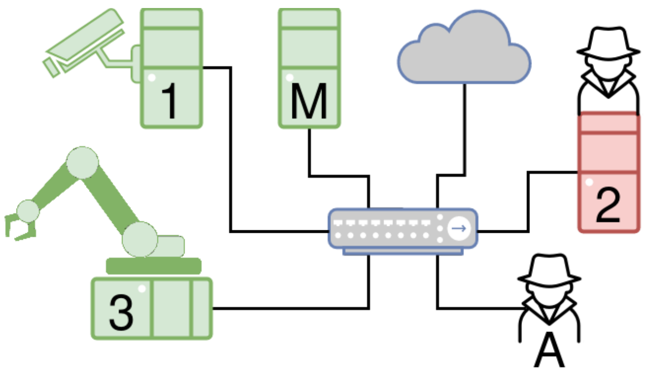

Fig. 1. Sketch of an exemplary ROS setup. Multiple hosts are connected in a network, a connection to an office network (depicted as a cloud) is shown. Two attackers are connected to the network.

security issues overlaps with other works by the same authors [11], [12], which will be discussed in section VII.

Previous works cover different aspects of ROS security, such as performance impact or security concerns outside of the network, yet none of them describes a structured identification of requirements for mechanisms that protect data transmitted over a network in between ROS nodes. Many protection mechanisms for ROS have been proposed, but no structured evaluation and comparison has been published.

IV. Scenario

A typical scenario in which ROS could be used would be a robot equipped with a gripper, which has to perform a pick and place task. For this purpose, the system is equipped with a camera, which detects the objects to be picked up. We depicted a simplified version of this scenario in Figure 1, where the robotic arm is operated by ROS nodes running on one host (labeled “3”) and the camera is run on a different host (labeled “1”). A ROS master is present (labeled “M”). An image processing and path-planning algorithm is run on a more powerful fourth host (labeled “2”). The hosts are connected via an ordinary network. To receive data on which tasks to perform, the entire setup is connected to an office network (depicted as a cloud).

V. Attacker Models

We model two realistic network based attackers that threaten ROS based systems. Based on these models, we then identify security objectives a protection mechanism must fulfill in order for the ROS to be considered resilient.

A. *Mitm Attacker*

A Man-in-the-Middle (mitm) attacker is a common network-based attacker model based on the established Dolev-Yao attacker model [13]. He is, either through his networking position or through techniques such as Address Resolution Protocol (ARP) spoofing, in a position where all network packets flow through him. The attacker is able to read the data transmitted, modify the packets before forwarding them to the intended recipient or drop them completely. He is not able to break encryption or other cryptographic protection mechanisms. The mitm attacker is not able to manipulate or compromise legitimate ROS nodes, but able to set up rogue ones.

The presence of a mitm attacker is realistic in many scenarios, as ROS systems rarely operate completely isolated. An attacker could connect to the network via an unprotected physical network port or compromise a system that is independent of the ROS application but connected to the same network, e.g. a vulnerable desktop computer. An example for mitm attackers is depicted in Figure 1. The attacker (labeled "A") managed to connect to the network the ROS operates in. Alternatively, the mitm attacker might have been able to connect from the office network (depicted as a cloud).

B. *ROS Node Attacker*

The ROS node attacker extends the mitm attacker modeled above, as he is able to compromise a legitimate ROS node. He thus has access to all resources available to the node such as cryptographic keys, network topology information and the nodes network access, which he can use to send and receive information using the nodes identity.

The existence of a ROS node attacker is realistic as ROS nodes are commonly operated on standard operating systems that could be compromised through a security vulnerability independent of ROS. In addition, ROS is occasionally used in heterogeneous environments where nodes are operated by different entities and cannot trust each other. An example for a ROS node attacker is depicted in Figure 1, where the attacker managed to compromise the compute node of the setup (labeled "2").

VI. Evaluation Criteria

Based on the scenario described in IV and the attackers described in V, we identified criteria that protection mechanisms will be evaluated against. The first four criteria are based on general security objectives that apply to the ROS context and the scenario given. Following are criteria that were identified for the specific context of ROS based systems.

A. *Confidentiality of Communication*

Information transmitted in a ROS network can contain sensitive information such as intellectual property or company secrets, e.g. the speed at which a robot is operating. Additionally, an attacker could use information he learns to perform further attacks on the ROS. The confidentiality of the information transmitted in the ROS thus needs to be protected. In many cases, encryption can be used to do so.

B. *Confidentiality of Communication Metadata*

ROS topics are often given names that can reveal some information about their content, such as "emergency-power-off-switch". Using this information, an attacker may be able to deduce some parts of the inner workings of a ROS, which can either be a company secret by itself or help him to perform additional attacks. Metadata that may be worthy of protection includes: ROS topic names, ROS node names and the access control policies. Hiding the amount of data transmitted is challenging and will not be considered in this work.

C. *Integrity and Authenticity of Communication*

Many ROS nodes act upon information received from other nodes via the communication network. If this information is tampered with or if illegitimate information is injected by an attacker, the receiving node may perform unwanted or even physically dangerous actions. In most cases, under the assumption of realistic attacker models, tamper-proof communication systems are not available. Instead, by employing cryptographic means, it is possible to build communication systems that enable the receiving node to decide if information has been tampered with in transit. Additionally, these means enable the receiving node to authenticate the sender (verify its identity) to make sure, the information is sent by a legitimate node.

D. *Access Control*

Mechanisms that protect the confidentiality and integrity of communication need to be able to differentiate in between legitimate nodes and attackers. The simplest way is to consider all legitimate ROS nodes that can be uniquely identified (e.g. by a cryptographic key) as trustworthy. In the presence of an attacker that manages to compromise even a single ROS node (see section V-B), this approach fails.

Access control measures can be used to limit a node to be only able to publish and subscribe to those ROS topics it needs to read and write for its legitimate operation. If access control measures are implemented, an attacker who gains control of a node is bound by those limits, thus his impact on the system is limited. Access control can be implemented based on different identifying criteria, e.g. the possession of a cryptographic key or IP address. Also, it can be applied on different levels, such as per ROS node or per host.

E. *Latest ROS Release and Last Contribution*

Many of the protection mechanisms interact closely with ROS, making it likely that a mechanism might not be compatible with ROS releases it was not explicitly designed for. In case a ROS environment is updated to the latest ROS release, the protection mechanism will most likely also need to be updated to a compatible newer release. Additionally, most software contains bugs and vulnerabilities that are found over time and need to be resolved. It is thus important to choose a protection mechanism that is under active development. Even though this is not a guarantee for new releases and patches, projects that have been abandoned are less likely to provide releases compatible with future ROS releases. In this criterion we thus check, for the latest ROS release a protection mechanism is compatible with and check when the latest relevant contribution to the protection mechanism was made.

F. *Key Exchange Requirements*

As described in section I-C, most protection mechanisms rely on previously exchanged cryptographic secrets. They should make it as simple as possible to perform these key exchanges whilst not relying on insecure methods. In most cases, separate trusted channels need to be operated manually. It is thus relevant, how many key exchanges are needed, depending on the number of ROS nodes, to set up the system.

VII. Evaluation of Protection Mechanisms

In order to be able to uphold the security requirements identified in VI, the ROS community has come up with various ideas and concepts. We describe the idea of each relevant protection mechanism, analyze, which evaluation criteria it fulfills and state identified problems.

A comparison of the mechanisms is shown in Table I.

A. *ROS without Protection Mechanisms*

ROS as originally presented in [1] does not provide any security measurements relevant to the network communication [5], [14], [17]. Without additional mechanisms, ROS 1 does not meet any of the security requirements. The newest release was published in Mai 2020.

B. *Secure Robot Operation System (SROS)*

SROS [14] uses Transport Layer Security (TLS) to protect the ROS communication. To use SROS, the operator must set up a Public Key Infrastructure and generate X.509 certificates for each ROS node.

SROS does protect the confidentiality and integrity of the communicated data as well as providing access control on a per-node basis using the cryptographic X.509 certificates. Because the X.509 certificates used for transport security and access control are transmitted in plain, the access control policies are not kept confidential. We thus consider SROS not to protect metadata confidentiality. To equip each node with a X.509 certificate, one cryptographic key must be exchanged in between each node and the PKI. As of November 2020, the SROS guide was outdated and the last relevant commit to SROS on github[1] was from 2016. It can thus be assumed that SROS is not maintained or developed any more.

C. *SRI Secure ROS*

SRI Secure ROS was developed as a fork of the ROS core packages. It uses IPSec in the VPN concept (see section VII-K) to encapsulate the whole communication.

By utilizing a VPN, SRI Secure ROS fulfills the requirements for confidentiality, integrity and metadata confidentiality as all ROS communication data is completely wrapped. Secure ROS implements access control on a per-host basis using the ip-addresses as identifier, thus all nodes running on the same host share the same privileges. The cryptographic keys for the VPN should be generated on one host and distributed to the other hosts, thus requiring only one key exchange per host. The code for Secure ROS is available on github [2], the documentation is public[3] but neither has significantly changed since April 2017, no release for current ROS release is available. It can thus be assumed that Secure ROS is not maintained or developed any more.

D. *ROS-AES-Encryption*

ROS-AES-Encryption (Advanced Encryption Scheme) as proposed in [5] is based on an idea described in [4]. For each ROS node p that publishes to topic $/topic/messages$, a secondary node p_{crypt} is created on the same host that reads the unencrypted data from $/topic/messages$ and republishes it in encrypted form to $/topic/encrypted/messages$ for other nodes to read. For any receiving ROS node r, the reverse process is applied: A node r_{crypt} is created on the same host that subscribes to $/topic/encrypted/messages$, decrypts the messages received and republishes them to $/topic/messages$ for the node r to read.

Assuming that Galois/Counter Mode (GCM) or a similar authenticated encryption block mode is used for Advanced Encryption Scheme (AES), this concept not only protects the confidentiality but also the integrity of the data transmitted. As only the content of the ROS communication is encrypted, metadata such as topic names is not protected. ROS-AES-Encryption is proposed as a rough concept, no implementation is publicly available. It is unclear if all ROS topics should use the same AES key or if individual AES keys should be used per topic.

1) *Identical AES Key:* By using the same AES key across all ROS topics, no access control is provided. On the other hand, only a single key needs to be distributed to all ROS nodes, requiring one key exchange per node.

2) *Individual AES Keys per topic:* By using individual keys per topic, a limited form of access control is established as only those ROS nodes with access to the key can publish or subscribe to the topic. On the other hand, the effort for distributing the keys is significantly larger. The number of key exchanges is given by: $\sum_{i=1}^{t} n_{T_i}$, where t is the amount of topics, T_i an individual topic and n_{T_i} the amount of nodes participating in a Topic T_i Due to the fact that AES is a symmetric encryption scheme, all ROS nodes who have access to the key can both publish and subscribe to the corresponding topic, the level of access control that can be provided is thus limited.

E. *Application-level Security for ROS-based Applications*

In [11], the authors describe a protection mechanism that uses X.509 certificates to identify nodes and a central authentication server that manages the symmetric keys needed to access the content of the ROS topics. Any node that wishes to publish or subscribe to a topic needs to obtain the symmetric encryption key for said topic from the central authentication server. For publishing nodes, the authentication

[1] https://github.com/ros/ros_comm/tree/sros

[2] https://github.com/SRI-CSL/secure_ros

[3] https://sri-csl.github.io/secure_ros/

TABLE I
EVALUATION RESULTS AND COMPARISON OF PROTECTION MECHANISMS

	Confidentiality	Metadata confidentiality	Integrity	Access Control	Latest ROS release supported	Last contribution [1]	Key exchanges [2]
ROS 1	–	–	–	–	n/a	10/20	n/a
SROS [14]	X	–	X	X ^{N,C}	kinetic	08/16	n
SRI Secure ROS	X	X	X	X ^{H,I}	kinetic / lunar	04/17	h
ROS-AES Encryption (no AC) [5]	X	–	X ^{3}	–	n/a	(no code released)	n
ROS-AES Encryption (AC) [5]	X	–	X ^{3}	X $^{N, C}$	n/a	(no code released)	$\sum_{i=1}^{t} n_{T_i}$
A.-l. Security for ROS [11]	X	–	X	X $^{N, C}$	n/a	(no code released)	n
Secure comm. for. ROS [12]	X	?[4]	X	X ^{N,C}	n/a	(no code released)	none [5]
ROSDN [15]	–	–	–	X ^{N,I}	n/a	08/18	n/a
ROSRV [16]	–	–	–	X ^{H,I}	kinetic	01/20	n/a
ROS 2	–	–	–	–	n/a	11/20	n/a
SROS 2 / DDS-Security	X	–	X	X ^{N,C}	foxy	11/20	n
ROS + Virtual Private Network (VPN) Concept	X	X	X	–	any	(concept)	unknown [6]

[1] As of November 2020 (Dates given as Month/Year)
[2] n / h is the amount of nodes / hosts respectively
[3] If a non-malleable cipher mode is chosen
[4] Depends if TLS $\geq$ 1.3 is used
[5] Key exchanges can be performed automatically
[6] Depends on configuration details
^{N} on a per-ROS node basis
^{H} on a per-host basis
^{C} based on cryptographic keys or certificates
^{I} based on the IP address or other networking features

server maintains a list of topics each publisher is allowed to publish to.

Applying this concept exclusively, thus ignoring all messages that do not follow the concept, will protect the confidentiality and integrity of messages using unspecified cryptographic algorithms. As it is applied on top of the standard ROS communication, metadata confidentiality, e.g. for node and topic names is not provided. Access control is described for publishing nodes only, but we assume that it should be equally applied to subscribing nodes. It is not described, how each nodes should be equipped with a certificate but most likely one key exchange per node will be required. According to the authors, the code for this concept has not been published.

F. Secure communication for the Robot Operating System

This concept was presented in [12] as an extension to the Application-level Security for ROS-based Applications as presented in section VII-E. It relies on the well-established TLS or Datagram TLS (DTLS) protocols (for TCP or UDP based communication, respectively) and X.509 certificates. Other than in the previously described concept, this extended version works on a peer-to-peer level and does not rely on a central node to manage different keys and access control. The X.509 certificates used contain the identity of the node and a list of topics each node is authorized to publish and subscribe to. The concept is designed to be included in the ROS communication by implementing it into the TCPROS / UDPROS handshake, instead of working on top of it. It proposes that the manufacturer should equip each device with a cryptographic key stored in a Trusted Platform Module (TPM) that should be used to simplify the installation of an operator-issued X.509 certificate.

By relying on standard techniques, this concept provides confidentiality and integrity to the communication. At the time this concept was published, certificates were transmitted in plain using TLS. If the newer version, TLS 1.3, is used, the confidentiality of the certificates is protected as well, thus protecting metadata confidentiality. Using the TPM, cryptographic keys for the production system can be exchanged without the need for manual interaction. According to the authors, the code for this mechanism has not been published.

G. ROSDN

ROSDN is a concept presented together with ROSWatch and ROS-Policy-Language as ROS-Defender in [15]. The idea is to learn the state of a ROS and then use¸ this learned state and pre-defined rules to configure Software Defined Networking (SDN) techniques to limit the communication in the network in a way that will allow all legitimate communication but prevent possible attackers from setting up new and potential harmful connections. To be able to use SDN techniques, special networking hardware is required.

ROSDN does not provide any confidentiality, metadata confidentiality or integrity protection mechanisms but focuses on access control. It analyzes, which open network ports belong to which ROS nodes and topics, thus access-control is provided on a node level, based on networking properties. ROSDN has been partly published [4], there have not been any recent changes, though. It can thus be assumed that ROSDN is not maintained or developed any more.

[4] https://github.com/seanrivera/ROSwall

H. ROSRV

ROSRV[5] [16] is a product that needs to be installed in a mitm position to the ROS master and is used to scan incoming ROS communication and decide whether to forward it to the ROS master. One factor that can be used to decide if packets should be forwarded to the ROS master is the host that sent them. Additionally, ROSRV can be used to verify the contents of the incoming ROS communication based on system states. One example given by the authors is that the request for the robot to shoot a gun will be dropped if the gun is currently pointed at the robot itself. This example implies that ROSRV needs to be adapted to the robot it is being used on to be able to keep track of system states and take decisions based on them.

ROSRV does not provide any confidentiality, metadata confidentiality or integrity protection mechanisms but focuses on access control based on IP addresses and plausibility checks against the system state. The source code for ROSRV publicly available and changes have been made recently.

I. ROS 2 without DDS Security Mechanisms

ROS 2 relies on DDS as a middleware that is responsible for the network communication.

Even though DDS can provide security mechanisms, they are disabled by default. ROS 2 on its own thus does not fulfill the security requirements for confidentiality, integrity, and access control. The newest release (as of November 2020) was published in June 2020.

J. SROS 2 / DDS-Security

DDS, on which ROS 2 relies for data transport, provides some optional security mechanisms defined by the OMG [18]. They rely on a PKI that issues identity certificates for each node. Using these certificates, AES-GCM secured connections can be established. Additionally, permission files can be created and signed by the Certification Authority (CA), which enable ROS 2 to limit nodes to publish and subscribe to specified topics only. SROS 2 is a set of tools designed to facilitate the use of the security mechanisms provided by DDS.

The DDS Security mechanisms can be used to provide confidentiality and integrity to the transmitted data and to create and enforce access control on a per-node level, based on cryptographic certificates. In its current release [6], the DDS Security initialized with SROS 2, transmits access control policies including node and topic names in plain text. It does thus not provide metadata confidentiality. By employing certificates and a PKI, only one key exchange per node is required. SROS is under active development, the DDS security extensions are integrated in most of the DDS implementations and should be supported in the future.

[5]Website for the project at http://fsl.cs.illinois.edu/index.php/ROSRV
[6]tests were performed on ROS 2 release "Eloquent Elusor"

K. ROS + VPN

VPNs are designed to encapsulate generic network traffic and provide protection to the data transmitted. This can be used in a ROS by setting up VPN connections in between all hosts and routing all ROS communication through them.

Using a VPN provides confidentiality, metadata confidentiality and integrity to the encapsulated ROS messages. Access control cannot be provided with this concept alone. There are different VPN implementations and different configuration options, thus the key exchange requirements cannot be generalized. VPNs can be applied to ROS 1 as well as ROS 2.

VIII. Discussion

Comparing the protection mechanisms presented in section VII and Table I, we see that none of them fulfills all of the evaluation criteria. It is thus necessary to carefully choose a mechanism depending on the exact scenario. The protection mechanism needs to be compatible with the ROS version used, the first criterion will thus be the ROS version.

A. Discussion for VPNs

ROS + VPN can be applied independently of the ROS version and release and satisfies confidentiality, metadata confidentiality and integrity. Access control, however, cannot be provided without ample extensions to the concept. Thus it cannot protect against an attacker that has compromised a ROS node (see section V-B). Because it is not required to be integrated into ROS (1 or 2), a VPN does not need to be updated alongside ROS and thus provides a stable protection mechanism that is not dependent on being adapted to changes that come with new releases of ROS.

As VPNs are general-purpose mechanisms, they must be carefully evaluated in the specific context. This is particularly important if they are used in special conditions, such as on low performance ROS nodes or in situations where hard real-time requirements apply.

B. Discussion for ROS 1

We have identified nine protection mechanisms compatible with ROS 1. Even though some of them have a concept that would be able to fulfill all the security requirements, none of them is ideal for practical use. Four protection mechanisms (*ROS-AES Encryption*, *Application-level Security for ROS-based Applications*, *Secure communication for the Robot Operating System* and *ROSDN*) are concepts described in academic publications, their implementations have not, or only in part, been publicly released. They are thus not ready to be used in a productive environment.

The three protection mechanisms with publicly available implementations (*SROS*, *SRI Secure ROS* and *ROSRV*) are compatible with ROS release "Kinetic Kame" for which supported will be dropped in Mai 2021. *SRI Secure ROS* is also compatible with ROS release "Lunar Loggerhead", which ran out of support in Mai 2019. Although it is possible that they still work with more recent ROS releases, this is not officially supported by the maintainers. It is unclear if any of the three protection mechanisms will be released for newer releases of ROS. Because the last contribution to *SROS* and *SRI Secure ROS* has been made in 2016 and 2017, respectively, this seems to be unlikely for at least those two.

Due to the downsides of all of the protection mechanisms, it is unclear, how a setup based on ROS 1 should be operated securely. Setting up a VPN is most likely the best choice in most scenarios, however this does not fulfill all requirements as previously identified. Also, setting up a VPN in a correct and secure way does require some knowledge on networking and requires the administrator to manually handle cryptographic keys. A set of tools that simplify the setup procedure and the necessary configuration of ROS would be helpful to ROS operators less experienced with VPNs and network settings. The development of ROS 2 should thus be closely monitored and a migration should be performed as soon as ROS 2 fulfills all requirements of the individual scenario.

C. Discussion for ROS 2

Only *SROS 2 / DDS-Security* and the *ROS + VPN* concept are compatible with ROS 2.

DDS-Security does not provide metadata confidentiality. As described above, *ROS + VPN* does not provide access control. It depends on the specific situation which security requirement is more important; both techniques can be combined in order to compensate for the shortcoming of the other. Due to the availability of the *SROS 2* tooling provided by the ROS community, activating the DDS security extensions is easier and requires less networking expertise than configuring *ROS + VPN*.

IX. Conclusion and Future Work

In this work, we identify realistic network-based attacker models for ROS based applications. We describe security objectives that need to be upheld to protect against those attackers and other requirements protection mechanisms need to fulfill in order to be usable in a productive environment. For both ROS 1 and 2, we describe in short how the available and conceptually described protection mechanisms work and evaluate them against the previously defined criteria. We show the results of our evaluation in a comprehensive table for quick reference. Because none of the the protection mechanisms fulfill all of the requirements, we discuss their applicability for ROS 1 and 2, and give advice on which protection mechanisms to use for either version.

Multiple academic works exist that compare the performance of a subset of the protection mechanisms in simplified scenarios (usually using only two ROS nodes), yet there is no comparison of all mechanisms in multiple larger, realistic scenarios. Additionally, the shortcomings of at least one of the protection mechanisms for both ROS 1 and 2 need to be mitigated in order to provide a mechanism, that protects all aspects of ROS-based applications.

Acknowledgment

This research was funded by the Ministry of Economic Affairs Baden-Württemberg as part of the CyberProtect project.

References

[1] M. Quigley, et al., "ROS: an open-source Robot Operating System," *ICRA Workshop on Open Source Software*, vol. 3, Jan. 2009

[2] Object Management Group, "Data Distribution Service," *Object Management Group*, 2015. [Online] Available: https://www.omg.org/spec/DDS/1.4 [Accessed Nov. 15, 2020]

[3] N. DeMarinis, S. Tellex, V. P. Kemerlis, G. Konidaris, and R. Fonseca, "Scanning the Internet for Ros: A View of Security in Robotics Research," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8514–8521, May 2019

[4] F. J. R. Lera, et al.J. Balsa, F. Casado, C. Fernández, F. M. Rico, and V. Matellán, "Cybersecurity in Autonomous Systems: Evaluating the performance of hardening ROS," *Workshop on physical Agents*, p. 47, 2016

[5] D. Portugal, M. A. Santos, S. Pereira, and M. S. Couceiro, "On the Security of Robotic Applications Using ROS," *Artificial Intelligence Safety and Security*, 1st ed., p. 444, 2017

[6] J. Kim, J. M. Smereka, C. Cheung, S. Nepal, and M. Grobler, "Security and Performance Considerations in ROS 2: A Balancing Act," *arXiv: 1809.09566*, Sep. 2018

[7] G. W. Clark, M. V. Doran, and T. R. Andel, "Cybersecurity issues in robotics," *2017 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*, pp. 1–5, Mar. 2017

[8] V. DiLuoffo, W. R. Michalson, and B. Sunar, "Robot Operating System 2: The need for a holistic security approach to robotic architectures," *International Journal of Advanced Robotic Systems*, no. 3, May 2018

[9] V. DiLuoffo, W. R. Michalson, and B. Sunar, "Credential Masquerading and OpenSSL Spy: Exploring ROS 2 using DDS security," *arXiv: 1904.09179*, Apr. 2019

[10] 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, Dec. 2017

[11] B. Dieber, S. Kacianka, S. Rass, and P. Schartner, "Application-level security for ROS-based applications," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4477–4482, Oct. 2016

[12] B. Breiling, B. Dieber, and P. Schartner, "Secure communication for the robot operating system," *2017 Annual IEEE International Systems Conference (SysCon)*, pp. 1–6, Apr. 2017

[13] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, Mar. 1983

[14] R. White, D. H. I. Christensen, and D. M. Quigley, "SROS: Securing ROS over the wire, in the graph, and through the kernel," *arXiv:1611.07060*, Nov. 2016

[15] S. Rivera, S. Lagraa, C. Nita-Rotaru, S. Becker, and R. State, "ROS-Defender: SDN-Based Security Policy Enforcement for Robotic Applications," *2019 IEEE Security and Privacy Workshops (SPW)*, pp. 114–119, May 2019

[16] J. Huang et al. "ROSRV: Runtime Verification for Robots," *Runtime Verification*, vol. 8734, pp. 247–254, 2014

[17] D. Portugal, S. Pereira, and M. S. Couceiro, "The role of security in human-robot shared environments: A case study in ROS-based surveillance robots," *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 981–986, Aug. 2017

[18] Object Management Group, "DDS Security," *Object Management Group*, 2018. [Online] Available: https://www.omg.org/spec/DDS-SECURITY/1.1 [Accessed Nov. 15, 2020]