IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 11 May 2023, accepted 24 May 2023, date of publication 1 June 2023, date of current version 7 June 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3282169*

RESEARCH ARTICLE

# An SDR-Based Cybersecurity Verification Framework for Smart Agricultural Machines

**ROBERTO CAVIGLIA, (Student Member, IEEE), GIOVANNI GAGGERO, (Member, IEEE), GIANCARLO PORTOMAURO, FABIO PATRONE, (Member, IEEE), AND MARIO MARCHESE, (Senior Member, IEEE)**

Department of Electrical, Electronics and Telecommunications Engineering and Naval Architecture (DITEN), University of Genoa, 16145 Genoa, Italy

Corresponding author: Giovanni Gaggero (giovanni.gaggero@edu.unige.it)

This work was supported in part by the ECSEL Joint Undertaking (JU) through the European Union's Horizon 2020 Research and Innovation Program and Austria, Czech Republic, Germany, Ireland, Italy, Portugal, Spain, Sweden, Turkey, under Grant 876852.

**ABSTRACT** The agricultural sector increasingly makes use of automated and/or remotely-controlled machines to improve performance and reduce costs. These machines, called Smart Agricultural Machines (SAMs), integrate different information and communication technologies for monitoring and control purposes and can be remotely controlled by using proprietary protocols. This makes it difficult to assess the vulnerabilities of the system, in particular for non-proprietary-parties. SAMs are cyber-physical systems often employing private protocols and can be objects of attacks. In this context the paper proposes a framework, based on Software Defined Radio (SDR) technology, for cybersecurity verification of SAMs, in order to fill the gap in the state of the art since no technical standard specifically addresses cybersecurity in this environment; the paper describes the testbed developed and exploited to show the effectiveness in detecting vulnerabilities and assessing the SAM security, in particular focusing on the wireless communication channels, and reports the obtained results.

**INDEX TERMS** Smart agriculture, autonomous machines, cybersecurity, software defined radio (SDR), wireless communications, penetration test.

## I. INTRODUCTION

Many sectors increasingly rely on information and communication technologies for monitoring and control purposes. One of them is certainly the agricultural sector which makes use both of sensor networks and of autonomous or remotely-controllable machines [1]. In particular, Smart Agricultural Machines (SAMs) are cyber-physical systems since their safety and security depend on the correct behavior of mechanical and information/communication devices. SAM remote control networks show similarities with the ones used in the automotive environment. For example, many SAMs exploit CANBus as the communication protocol between Electronic Control Units (ECU), engines, and sensors. CANBus suffers from severe vulnerabilities in terms of cybersecurity but it is still widely used since these systems are considered isolated networks so the only way to threaten the system is to gain physical access to the network. Unfortunately, this assumption can be hardly considered still valid because most SAMs use different communication technologies to be remotely monitored and controlled. These technologies include IoT communication protocols [2], [3] and the Global Navigation Satellite Systems (GNSS), but also protocols specifically developed for SAMs. In these cases, if the specifications of the communication protocols are unknown to non-proprietary parties, it is very hard, if not impossible, to perform a vulnerability assessment. While cybersecurity for autonomous vehicles has been broadly investigated in the automotive sector [4], taking into account different use cases, such as truck-trailer systems [5], the same cannot be said for sectors that inherit some communication technologies but that also present their own peculiarities, such as SAMs. The rationale behind this work is the lack of proper procedures and

The associate editor coordinating the review of this manuscript and approving it for publication was Lorenzo Mucchi.

guidelines to assess the cybersecurity level of these devices, in order to perform a risk assessment.

This paper proposes a framework aimed at allowing companies and non-proprietary parties to check the cybersecurity level of remotely-controlled SAMs based on unknown communication protocols. The proposed procedure exploits the capabilities of Software Defined Radio (SDR) technology to find out the vulnerabilities of the control protocol assuming the use of an unknown communication protocol, and so assess the system security level. The verification framework is based on a set of attacks with increasing complexity, so that it is possible, during the assessment phase, to evaluate the risk level. A testbed aimed at evaluating the effectiveness of the proposed framework is built by using off-the-shelf components. The ideas proposed and tested in this paper may be considered as a foundation to develop policies, guidelines, and procedures to handle cybersecurity issues for SAMs since, as discussed in the following, no technical standard specifically addresses cybersecurity in this environment. The proposed framework can also be contextualized within the multidimensional framework for verification and validation of automated systems safety and security proposed by the VALU3S project [6].

The paper is structured as follows. Section II discusses the reference literature concerning vulnerability assessment of wireless environments. Section III shows the typical architecture of a SAM control network also evidencing the main features as well as the related regulations. Section IV describes the proposed framework by detailing possible attacks with different complexity levels: jamming, replay, reverse engineering-based, and complex attacks such as a combination of the previous ones. Section V presents the developed testbed to evaluate the effectiveness of the proposed cybersecurity assessment framework. Section VI shows the results and Section VII reports the conclusions.

## II. STATE OF THE ART REGARDING VULNERABILITY ASSESSMENT FOR WIRELESS ENVIRONMENTS

Wireless devices are currently employed in many fields including industrial control systems and SAMs. The use of wireless technology gives great advantages in terms of the system's efficiency and management costs, but also extends the attack surface and risk, aspects not always properly taken into account when a system is designed. In order to assess the vulnerabilities of wireless environments by generating targeted attacks, the ideal solution would be to have dedicated radio interfaces for each available technology so to generate a large set of proper solicitations and tests by exploiting the right frequencies and modulations for (possibly all) off-the-shelf solutions. A possible cheaper chance is to bring SDR to the penetration testing community [7]. SDR allows receiving and transmitting signals personalizing multiple parameters, such as the employed modulation, and retrieving the data stream up to the bit for each received packet by using a single hardware component and open-source software. Applied to our context, SDR allows passively and actively attacking different wireless communication protocols and technologies. In this context, [8] proposes a preliminary framework for network security verification of automated vehicles in the agricultural domain and highlights the wireless interfaces of the control network that should be taken into account while performing vulnerability assessment.

The most common threats to wireless communication are carried out through jamming attacks. Concerning this issue, [9] proposes a possible taxonomy and different attack, defence, detection and prevention techniques structured for different types of wireless networks. Reference [10] proposes a survey on existing jamming attacks and anti-jamming strategies in wireless networks such as local area networks (WLANs); cellular and cognitive radio networks (CRNs); ZigBee, Bluetooth, vehicular, LoRa, and RFID networks; as well as Global Positioning System (GPS), millimeter-wave (mmWave), and learning-assisted wireless systems. An efficient attack methodology is based on reverse engineering that can lead to the complete control of SAMs in case of insecure protocols. Reference [11] proposes a survey of automatic protocol reverse engineering tools, summarizing and organizing them by the used approach, and focusing primarily on high-level protocols in packet-switched networks. Reference [12] focuses on wireless protocols and proposes a framework for automatic reverse engineering. The algorithm proposed in [12] is available as open-source software as part of the Universal Radio Hacker [13]. Reference [14] utilizes SDR to reverse engineer the communication protocol of a RFID public transportation card and shows how to capture tag-reader communications, access private information, and emulate both tags and readers. In the agricultural domain, [15] defines the requirements for cybersecurity in agricultural communication networks by considering different use cases, but without targeting SAMs. Reference [16] proposes a testbed for cybersecurity assessment of SAMs, but does not provide a practical approach for cybersecurity verification. As highlighted in [17] the digital agriculture is still at an early stage, and therefore there is no security framework developed explicitly for this environment. To the best of our knowledge, no scientific papers are specifically focused on the wireless remote control of Smart Agricultural Machines by taking into account safety requirements.

## III. CYBERSECURITY VERIFICATION OF AGRICULTURAL AUTOMATED VEHICLES

### A. STANDARDS AND REQUIREMENTS

There are several regulations regarding safety for remote controllers in the industrial environment, such as:

- IEC 60204-32, which provides requirements and recommendations concerning the electrical equipment of hoisting machines aimed at increasing people and asset safety, consistency of control response, and ease of maintenance.
- ISO 13849, which provides safety requirements and guidance about the design and integration of safety-related parts of control systems.

- IEC 60950-1, which defines the basis for the safety of information technology equipments.
- IEC 61000-6-2, which provides EMC (Electromagnetic Compatibility) immunity requirements that apply to electrical and electronic equipment intended for use in industrial locations in the frequency range 0-400 GHz.

Although ISO/IEC 62443-4-1 specifies process requirements for the secure development of industrial automation and control systems. No regulations and standards define the approach to test cyber risk on SAMs. In case of devices installed in cars, approved tractors and trucks, the new regulation UNECE R155 - Cyber security and cyber security management system is the reference that all OEMs (Original Equipment Manufacturers) have to comply with. Standard ISO/SAE 21434 sets guidelines to secure high-level processes in connected cars. Nevertheless, these standards are referred only to the automotive sector and do not consider vehicles with similar features, such as SAMs, but not designed to operate on public roads. Additionally, it is worth remarking that ISO/IEC 62443-4-1, ISO/SAE 21434 and other standards do not define a framework for the validation and verification phases against cyber threats. For this reason, it is important to create a framework to help the manufacturers assess their products from the cybersecurity viewpoint to improve the quality and reliability of the product.

### B. REFERENCE ARCHITECTURES

The reference architecture of the SAM considered in this paper is composed of the following parts, as shown in Figure 1:

- Transmitter: it is a device, usually a joystick, that allows controlling the SAM remotely through direct communication with the receiver by using a proprietary protocol.
- Receiver: it is the SAM component in charge of translating the commands received from the transmitter to the control network by exploiting two asynchronous communication channels.
- Electronic Control Unit (ECU): it is the tool connecting the receiver, sensors and actuators, other communication means, and control network by using wired protocols such as CANBus. It is composed also of secondary ECUs.
- Sensors and Actuators: devices that generate data or perform their actions after receiving proper commands from the main ECU through the main communication protocol or directly with current and voltage signals.
- Other Communication interfaces: the SAM can receive different signals, including GNSS or, even if less frequently, cellular communications or IoT.

As discussed in [8], the attack model for SAMs is mostly related to wireless interfaces. The vulnerabilities of wireless communication significantly vary with the employed solutions. This paper focuses on a subset of technologies for the transmission of time-critical control-related information between a single controller and the vehicle.

Referring to the architecture in Figure 1, the wireless communication portion presents peculiar features whose detail is important in the context of the paper. An important characteristic is that the wireless portion is designed to connect the remote controller with the SAM, or, at most, with a limited number of devices. Consequently, transmitter and receiver devices do not need to implement a full communication stack, such as a TCP/IP one, but rather encapsulate the information directly in the lower layer protocol. On one hand, this action reduces the number of attacks that can be carried out to the protocol acting in the wireless communication portion, but, on the other hand, it is worth remarking that, in cyber-physical systems, commands are safety-critical and a malicious control violating the integrity of a command message can threaten the overall agricultural work or even human safety. Moreover, due to the simplicity of the protocol and the supposed reduction of the attack perimeter mentioned above, the communication protocol may have been designed without reference to cybersecurity risk.

More specifically, it is important to highlight that transmitted information on the communication link between transmitter and receiver contains commands for a cyber-physical system. Consequently:

- The lack of communication is very risky. Information needs to arrive safely because it is related to the capability of the SMA to switch to a safe operational mode. For example, if a command that requires the engine to be switched off does not arrive, a dangerous event may happen.
- Data integrity is extremely important. For example, if the command "turn right" is interpreted as a "turn left", potentially dangerous events may happen.
- Information is time-critical. Communication delays may impact the safety of the operation.
- The violation of confidentiality, even if not recommendable because it may provide information about the control process, does not represent a major threat in this environment.

The proposed framework has been designed for this context.

## IV. PROPOSED FRAMEWORK

In order to proceed with the vulnerability assessment, it is necessary to retrieve some communication information, such as the frequency used for the radio link and the communication mode (half-duplex or full-duplex). This information is usually directly reported on the devices but can anyway easily be retrieved by monitoring the energy of the transmission by using general-purpose software. Once established the range of frequencies, it is possible to proceed with the tests of attack of increasing complexity. We distinguish the possible attacks into four main categories: 1) denial of service through jamming; 2) replay attack, to send malicious commands; 3) reverse engineering, to send malicious commands without being bound to the captured traffic; and 4) complex attacks that, by using a combination of the attacks mentioned above,

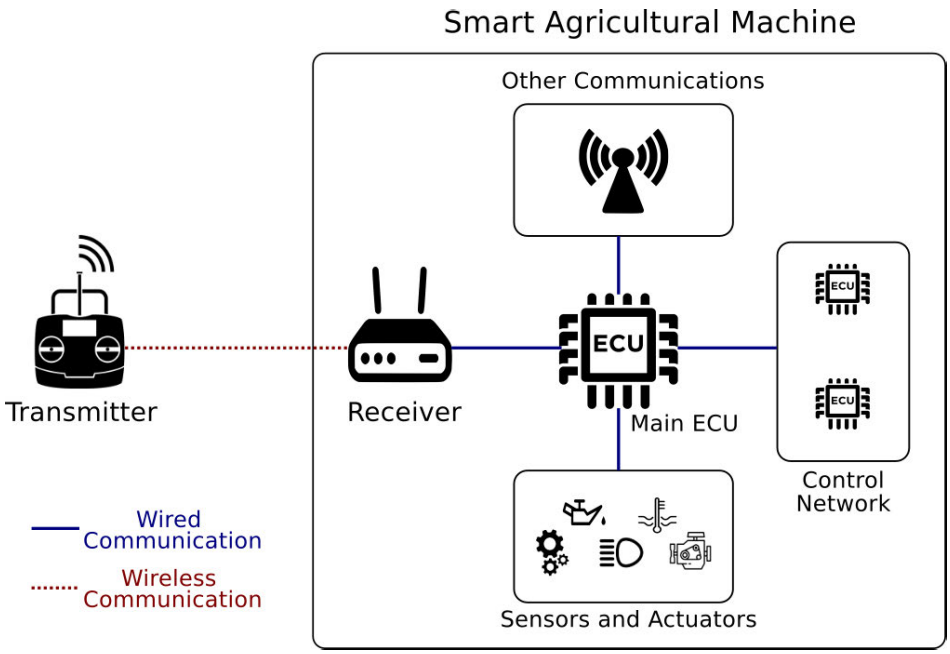


**FIGURE 1.** Architecture of a generic Smart Agricultural Machine.

can lead to a more sophisticated malicious control of the SAM. For each category details are provided in the following. Attack implementation is described in Section V-B.

### A. DENIAL OF SERVICE THROUGH JAMMING

The most common Denial of Service attack for wireless communication is the jamming attack. Jamming attacks aim to deny communication between the transmitter and receiver. These devices may or may not implement countermeasures for jamming attacks. While jamming attacks are almost always effective in wireless communications, in the risk assessment phase, it may be important to take into account the sophistication of the attack that is used to deny the communication.

#### 1) REGULAR JAMMER

Regular jammers are built by sending white noise either over a single fixed frequency, in case of half-duplex communication, or over a pre-defined set of frequencies, in case of full-duplex communication. Communication devices can be immune to this type of attack in case they implement some basic countermeasures such as frequency hopping mechanisms. Frequency hopping changes the communication frequency over a certain range in case the transmitter detects too much noise in the channel. Such a mechanism can be exploited also for safety reasons.

#### 2) RESPONSIVE JAMMER

Responsive jammers (also called reactive) are jammers that dynamically change the frequency over which they transmit white noise. In case the devices implement a frequency hopping mechanism, a reactive jammer is needed. Two cases can be distinguished: if it is possible to forecast the next frequency over which the transmission will be set (because the algorithm utilizes a predictable strategy), then the reactive jammer follows a pre-defined sequence. In case the algorithm utilizes a good mechanism to make the change, then it is necessary to develop an algorithm able to listen to the channel, detect the new frequency, and continuously "follow" the transmission. In this second case, time becomes crucial: if the jammer is too slow to change the frequency, the devices may be able to communicate by sending packets before the jammer manages to deny the communication.

#### 3) PACKET-BASED JAMMER

The two aforementioned attacks are based on the transmission of white noise over the channel. It is worth noticing that devices may behave differently in case they detects that the channel is occupied by white noise or by specific signals. Signals that are recognized as legitimate packets (for example, truncated packets) may lead the receiver to a different behavior. In this sense, a jammer built with legitimate-formed packets may succeed in denying the communication even without being responsive, because frequency hopping is not used by the receiver. In this case, the receiver continuously receives packets that are considered legitimate (and consequently the receiver does not switch the frequency), but the packets do not contain any real information and occupy the channel so preventing real packets to be received.

### B. REPLAY ATTACK

Replay attack aims to send malicious commands from a fake controller by re-transmitting previously captured signals.

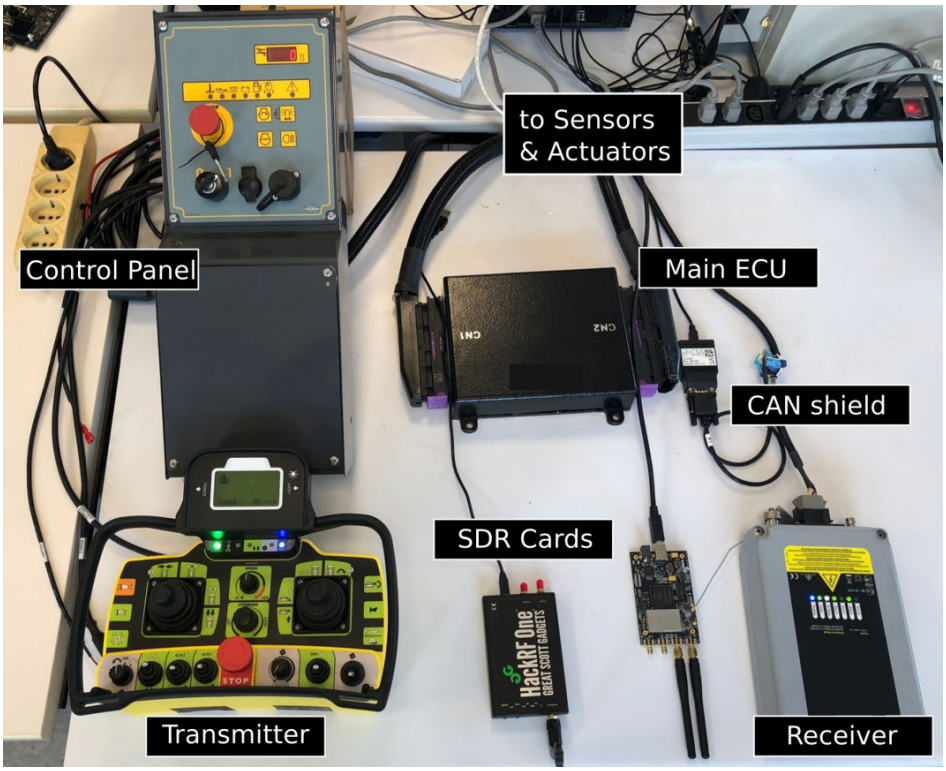


**FIGURE 2.** Developed testbed.

A replay attack is possible even without a complete knowledge of the protocol if the device does not implement any authentication mechanism.

#### 1) SIMPLE RETRANSMISSION

The most simple way to implement a replay is to capture the raw signal over a specific frequency, regardless of the interaction between transmitter and receiver. In this case, it is sufficient to replay the frames that contain the commands to perform the attack. If this attack is successful, it may be due to the fact that interacting devices do not implement authentication mechanisms and do not react to the transmission of packets by using acknowledgments as it happens, for example, in connectionless protocols.

#### 2) RECOGNIZED TRANSMITTER

The receiver may accept signals only after a negotiation phase, or subject to specific conditions of interaction between the controller and receiver as it happens in connection-oriented mechanisms. In this case, to replay a command, it is necessary to keep into account the interaction and possible acknowledgment exchanges.

#### 3) ABSENCE OF LEGITIMATE CONTROLLER

In addition to a connection-oriented communication, the receiver may not allow a substitution of the controller if there is another one already connected. In this case, the receiver accepts commands only if no other transmitter is already connected to it.

### C. REVERSE ENGINEERING

Reverse engineering aims to have a complete understanding of the protocol and its specifications.

#### 1) RAW DATA STREAM

As discussed in section III-B, protocols used in the wireless portion are usually raw data encapsulated directly over the physical or datalink layers, so it is not necessary to develop a full communication stack. Reverse engineering follows three main steps:

1) Recognition of the used modulation: this action allows translating the signal into a string of bits. The recognition can be done by recognizing the patterns of some parameters regarding the signal, such as the constellation diagram.
2) Structure of the frame: this action allows structuring the string of bits into the fields of a structured frame, term that, in this context, is used as equivalent to the term packet. Once defined the modulation, it is possible to define the boundaries and the composition of the packets.
3) Parsing of the protocol: to define the meaning of each field or group of bits within single frames. This can be achieved by comparing frames looking for similarities, or by using more complex and structured procedures.

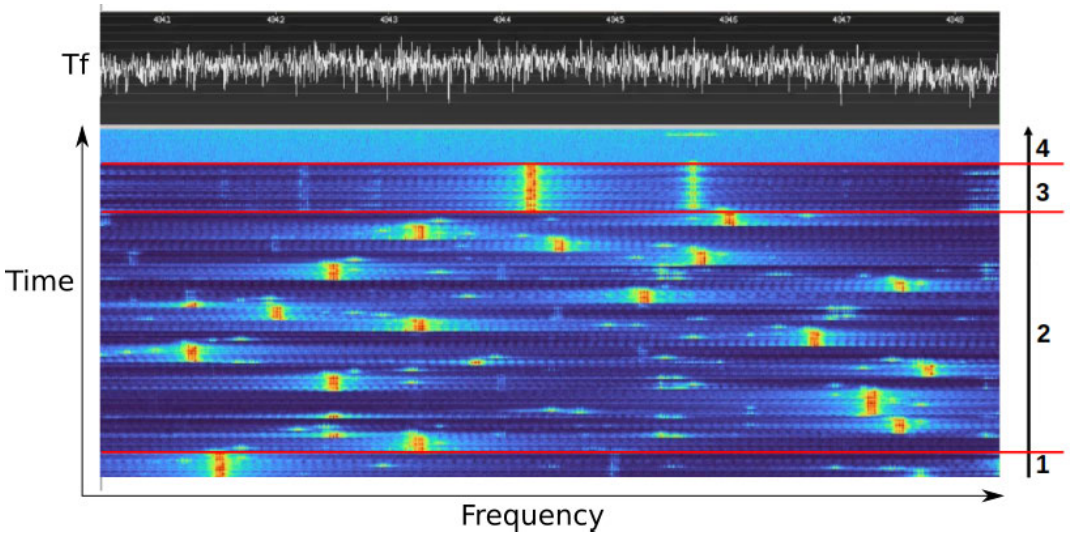


**FIGURE 3.** DoS Attack - GNU Radio visualization.

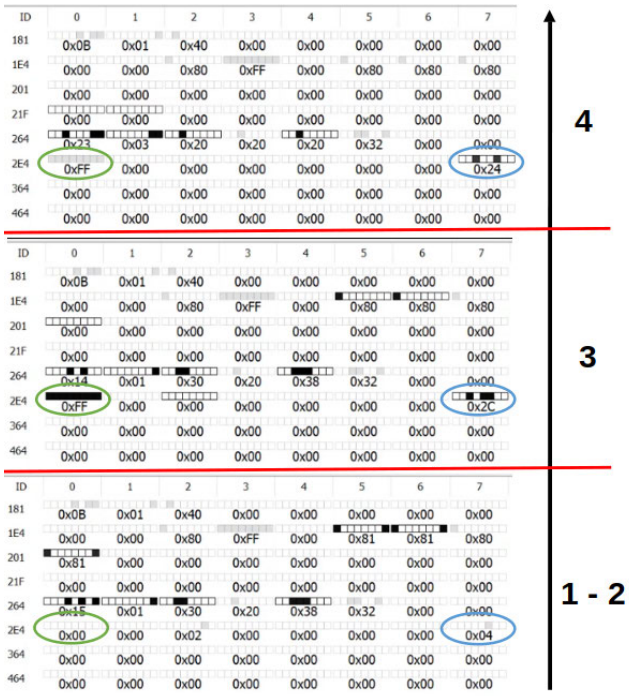


**FIGURE 4.** DoS Attack - CANBus visualization.

### 2) BREAK "SECURITY BY OBSCURITY"

The communication protocol may not implement cryptography between the transmitter and receiver. This choice is quite common since the protocol is "secured" by the fact that the details of the implementation are not revealed. Even if the frame is plaintext, it is not always possible to reverse-engineer the protocol. The wireless communication may not send the bit stream directly, but rather through a processed signal. One of the most common examples is the cyclic redundancy check (CRC): in this case, data get a short check value typically attached at the end of the frame, based on a specific math function. The calculation is repeated by the receiver and, in case the check values do not match, corrective action are taken against data corruption, for example discarding the whole frame. An attacker that can't retrieve the used CRC method is not able to send frames whose structure is acceptable by the receiver. This information can be retrieved in other ways, for example in case the attacker disposes of the transmitter hardware: if no authentication is implemented, it is not necessary to have information about the legitimate transmitter, but only about any other one that shares the "block diagram" of the transmission.

### 3) BREAK AUTHENTICATION—CRYPTOGRAPHY

In case of proprietary solutions, as said, it is uncommon that the developer chooses to encrypt the protocol. In case it does, due to economic and simplicity reasons, the solution may use cryptography schemes that are not totally secure, so that it could be theoretically possible to break the cryptography. The problem is how to do that with no a-priori knowledge of the protocol. As pointed out in [11], the use of encryption could make reverse engineering very hard, if not infeasible, but there may be some information about the protocol state machine that can be inferred even when data are encrypted. Further research has to be done in this field. In any case, it will be probably very hard to reverse engineer an encrypted protocol without any a-priori knowledge of the protocol itself.

## D. COMPLEX ATTACKS

We can also use a combination of the aforementioned attacks to gain a more precise malicious control of the system.

### 1) DELETING SELECTED COMMANDS

In case reverse engineering is successful, it could be possible to jam specific packets. For example, the attacker, being able

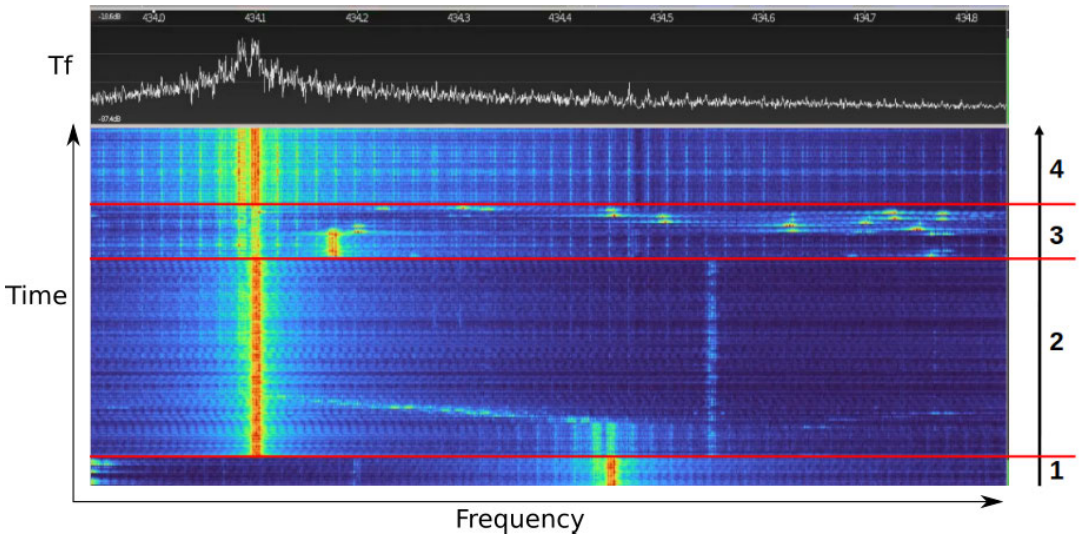


**FIGURE 5.** Replay Attack - GNU Radio visualization.

to eavesdrop packets, may recognize some patterns in the control and anticipate the control, jamming selected packets, even acting online.

#### 2) SUBSTITUTING THE CONTROLLER

If reverse engineering is successful and the communication has no authentication mechanism, it could be possible to jam the packets coming from the legitimate controller and send packets by using a fake controller. This attack would clearly represent a severe violation of the system safety.

## V. TESTBED

### A. ARCHITECTURE OF THE TESTBED

To test the proposed framework, we built a testbed composed of the control network of a real off-the-shelf SAM, which is a small tractor able to operate autonomously or remotely connected, as shown in Figure 2.

The control network is based on the CANBus protocol. There is a main ECU which identifies two CANBus segments: on one side, the ECU communicates with the control panel and with the various sensors and actuators by using either CANBus or direct voltage and current signals; on the other side, the ECU is connected to the receiver which communicates with the transmitter. The couple transmitter-receiver communicates by using a protocol of which we have no a-priori knowledge, except for the range of used frequencies. We used two main devices to interact with the control network: a CANBus shield and two SDR cards.

Through the CANBus shield, we could to monitor the CANBus, which is used in its basic form without authentication and encryption, and to eavesdrop packets. Since we did not dispose of the whole set of sensors and actuators, we checked the effects of the attacks by eavesdropping packets. For example, each time a replay attack was successful, the correspondent CANBus packet flowed into the network. The SDR card was used to implement the attacks. More details on the implementation are reported in the next section.

### B. IMPLEMENTATION OF THE FRAMEWORK

The framework has been put into practice primarily by using the open-source software GNU Radio [18]. GNU Radio is a free software development toolkit that provides signal-processing blocks to implement software-defined radios and signal-processing systems. It can be used with external RF hardware to create software-defined radios. The Hardware components include 2 main devices: the HackRF One [19] from Great Scott Gadgets, which is a Software Defined Radio peripheral to transmit and receive radio signals from 1 MHz to 6 GHz; the bladeRF 2.0 micro, which is a 2 $\times$ 2 MIMO, 47MHz to 6GHz frequency range, off-the-shelf USB 3.0 Software Defined Radio (SDR) [20].

In the case of jamming attack, we generated the white noise which is then transmitted over the channel by using the corresponding blocks in the GNU Radio graphical user interface. For the replay attack, we recorded packets by using a proper filter, saved them in a file on the PC, and re-transmitted them by building a simple GUI that allows dynamically setting the frequency and gain of the transmission.

## VI. PERFORMANCE EVALUATION

We carried out tests of increasing complexity basing on the framework described in Section IV. In the following sections we show the measures taken on the wireless channel thanks to the GNU Radio software and the measures taken on the CANBus network. The GNU Radio software shows two graph types: the first one is reported in the lower part of Figures 3 and 5) and shows the amount of transmission power by using different colors (the power is higher towards the red color) over frequency (abscissa) and time (ordinate); the second one is located in the upper part of Figures 3 and 5)


**FIGURE 6.** Replay Attack - CANBus visualization and Controller.

and shows the amount of transmission power in decibel (dBm) over frequency in a fixed instant, and more precisely on $T_f$, which is the last instant analyzed in the transmission power over frequency and time graph just described. Figures 4 and 6 show the CANBus variables: each variable is sent periodically on the bus and the figures show the state of these variables caught at certain times. The presence of the corresponding CANBus packets after the attack confirms its effectiveness.

### A. DENIAL OF SERVICE

During the first DoS attack test, we identify four main phases:

- Phase 1: Normal operation. The communication is regular and the machine is remotely controlled
- Phase 2: Jamming Attack - by using Regular and Responsive Jammers. We inject white noise over the channel frequency used for the communication and we start to slowly follow the frequency when it hops.
- Phase 3: Packet-Based Jammer: we perform the attack by injecting truncated packets and/or packets with white noise overlapped.
- Phase 4: we continue the attack started in phase 3 over time.

The effect of the attacks on the wireless channel is shown in Figure 3 after starting the jamming action. The couple controller-receiver operates a simple frequency-hopping mechanism over a range of 69 frequencies. The action performed by the regular jammer is totally ineffective. The sequence of frequencies is random so that is not possible to forecast the next frequency. The responsive jammer chases the frequency but it is too slow to effectively deny the communication. Figure 4 shows the state of CAN variables over time in the four phases. During the responsive jammer phase (Phase 2), the values don't change with respect to Phase 1 since the speed of the jammer chase is not sufficient. A simple regular jammer attack is not effective.

To prevent communication a packet-based jammer is more effective. We transmit a series of packets that have been previously recorded by truncating some packets or by adding white noise over the transmitted packet but always keeping a legitimated-formed packet. The receiver detects a busy channel but still recognizes the structure of the packets. Under such attacks, the receiver behaves differently with respect to the responsive jammer with white noise. As shown in Figures 3 and 4, the attack starts during phase 3: the receiver does not change the frequency and receives two signals. In the period of time corresponding to phase 4 the system crashes and the transmitter shuts down. As long as the attack carries on, the transmitter cannot re-establish the communication. At $T_f$, as shown in the graph in the upper part of Figure 3, there is no communication and only white noise is present over the frequency range used by the controller.

Looking at Figure 4, the information in the green circle takes the value $0x00$ if a controller is active and $0xFF$ if no controller is present. The one in the blue circle assumes different values during the different phases of the connection: during phases 1 and 2, the negotiation phase between the controller and receiver is finished and the value indicates a normal operation; during phase 3, the value indicates a re-negotiation attempt; during phase 4, the value indicates an error of no connected controller.

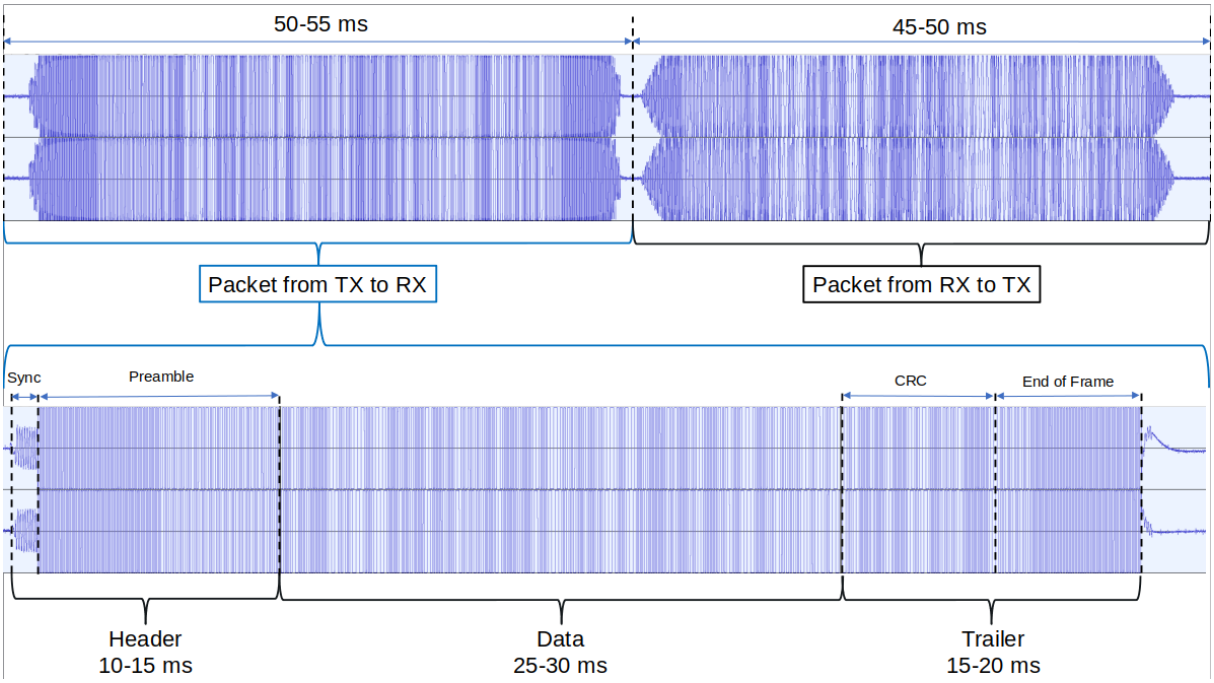


**FIGURE 7.** Visualization of the frame transmitted over the radio link.

Packet-based jamming was really successful. It is not the only dangerous and effective attack, a more sophisticated implementation of the responsive jammer may have worked but a packet-based jammer is probably simpler and less expensive from the implementation viewpoint.

### B. REPLAY

During the first Replay attack test, we identify four main phases:

- Phase 1: Normal operation. The communication is regular and the machine is remotely controlled
- Phase 2: Start of attack. The malicious signal overlaps the legitimate signal.
- Phase 3: Attack successful. The malicious signal substitutes the legitimate one. The machine is controlled by the malicious signal. The legitimate controller tries to re-establish the connection, but it fails.
- Phase 4: Substitution of the controller. The legitimate controller shuts down, so that the control is completely taken by the malicious signal.

The effect of the attacks on the wireless channel is shown in Figure 5. Figure 6 shows the impact on the CANBus network.

As clear in Figure 5, during phase 2, the malicious signal overlaps the legitimate one. During this phase, the receiver detects two signals. Two events can happen: 1) the receiver crashes, as in the jamming attack; 2) if the attacker manages to set the right transmission power, the malicious signal prevails and takes control of the SAM. The legitimate controller notices that it lost control by listening to the response frame and tries to re-establish the connection by repeating the handshake phase. If it fails, it shuts down. At this point, the malicious signal has a complete control of the SAM.

Looking at Figure 6, during phase 1, the information in the red circle contains a number that represents the speed of the machine. $0x81$ corresponds to speed equal to zero, while $0xFF$ corresponds to the maximum speed. Meanwhile, the information in the green circle contains the value $0x00$ if a controller is active and $0xFF$ if no controller is present. The green circle continuously contains the value $0x00$, also in Phases 3 and 4, which means that the malicious signal has completely substituted the legitimate one. In Figure 6, on the right, a photo of the controller during the different phases is reported. It shows what a human operator sees during the attack. The control of the SAM is completely lost in Phase 4.

This attack was successful in its simplest form: replay - single retransmission. The SAM does not implement any authentication mechanism, so we can forecast that it is possible to obtain a more sophisticated control of the SAM in case the attacker is able to reverse-engineer the protocol.

### C. REVERSE ENGINEERING

We also tried to reverse-engineer the protocol, so to be able to generate packets. By the analysis of the signal over time and some computed parameters, such as the constellation diagram, we could recognize the modulation used by the transmitter, which is GMSK. Also, we could identify two frames, which correspond to the upstream and downstream communication, regularly sent over the same frequency and so identifying a half-duplex communication. The next step has been the identification of the frame fields by looking for

similarities in corresponding packets over time, as shown in Figure 7

By looking at the portion of the frame that changes with respect to the commands sent by the transmitter, we identified the data field. Also, we could identify the bits corresponding to a specific command. Nevertheless, we could not generate packets that are recognized by the receiver. The reason is the CRC field: if we generate new packets by modifying portions of the data field but without modifying the CRC code, the receiver simply discards the packets. In this sense, even if we could reverse engineer the data field, we could not generate packets, because we could not replicate the CRC field without any a-priori information regarding the used algorithm. As discussed in Section IV-C, this information could have been retrieved by a deep analysis of the hardware, but this was beyond the scope of the present work. The ''security-by-obscurity'' protection of the wireless protocol cannot be considered a good solution for cybersecurity, but it can be considered sufficient for the risk assessment in specific contexts.

## VII. CONCLUSION

This work proposed a framework for the verification of cybersecurity of Smart Agricultural Machines (SAMs), and, in particular, of wireless communication, by exploiting the capabilities of software-defined radio technologies. The framework is based on attacks classified by an increasing level of complexity so that it is possible to evaluate the risk and choose proper mitigation strategies depending on the specific context. As discussed, few works in the literature specifically take into account the issue of cybersecurity of SAMs, and usually fail to provide practical approaches, so that results are hardly comparable to the state of art. We built a testbed in order to show the effectiveness of the proposed framework in assessing the vulnerabilities. Results show how commercial products may present severe vulnerabilities and remark the importance of having methods and tools for cybersecurity verification available. Future development in this field should be focused on standardization and on the definition of precise requirements that can be applied to a wide range of SAMs, in order to strengthen the cybersecurity of the whole agricultural sector.

## REFERENCES


[1] S. I. Hassan, M. M. Alam, U. Illahi, M. A. Al Ghamdi, S. H. Almotiri, and M. M. Su'ud, ''A systematic review on monitoring and advanced control strategies in smart agriculture,'' *IEEE Access*, vol. 9, pp. 32517–32548, 2021.

[2] B. B. Sinha and R. Dhanalakshmi, ''Recent advancements and challenges of Internet of Things in smart agriculture: A survey,'' *Future Gener. Comput. Syst.*, vol. 126, pp. 169–184, Jan. 2022.

[3] D. Glaroudis, A. Iossifides, and P. Chatzimisios, ''Survey, comparison and research challenges of IoT application protocols for smart farming,'' *Comput. Netw.*, vol. 168, Feb. 2020, Art. no. 107037.

[4] K. Kim, J. S. Kim, S. Jeong, J.-H. Park, and H. K. Kim, ''Cybersecurity for autonomous vehicles: Review of attacks and defense,'' *Comput. Secur.*, vol. 103, Apr. 2021, Art. no. 102150.

[5] X. Cai, K. Shi, K. She, S. Zhong, and Y. Tang, ''Quantized sampled-data control tactic for T-S fuzzy NCS under stochastic cyber-attacks and its application to truck-trailer system,'' *IEEE Trans. Veh. Technol.*, vol. 71, no. 7, pp. 7023–7032, Jul. 2022.

[6] J. A. Agirre, L. Etxeberria, R. Barbosa, S. Basagiannis, G. Giantamidis, T. Bauer, E. Ferrari, M. Labayen Esnaola, V. Orani, J. Öberg, D. Pereira, J. Proença, R. Schlick, A. Smrčka, W. Tiberti, S. Tonetta, M. Bozzano, A. Yazici, and B. Sangchoolie, ''The VALU3S ECSEL project: Verification and validation of automated systems safety and security,'' *Microprocessors Microsyst.*, vol. 87, Nov. 2021, Art. no. 104349.

[7] J.-M. Picod, A. Lebrun, and J.-C. Demay, ''Bringing software defined radio to the penetration testing community,'' in *Proc. Black Hat USA Conf.*, 2014, pp. 1–12.

[8] G. B. Gaggero, A. Fausto, F. Patrone, and M. Marchese, ''A framework for network security verification of automated vehicles in the agricultural domain,'' in *Proc. 26th Int. Conf. Electron.*, Jun. 2022, pp. 1–5.

[9] S. Vadlamani, B. Eksioglu, H. Medal, and A. Nandi, ''Jamming attacks on wireless networks: A taxonomic survey,'' *Int. J. Prod. Econ.*, vol. 172, pp. 76–94, Feb. 2016.

[10] H. Pirayesh and H. Zeng, ''Jamming attacks and anti-jamming strategies in wireless networks: A comprehensive survey,'' *IEEE Commun. Surveys Tuts.*, vol. 24, no. 2, pp. 767–809, 2nd Quart., 2022.

[11] J. Narayan, S. K. Shukla, and T. C. Clancy, ''A survey of automatic protocol reverse engineering tools,'' *ACM Comput. Surv.*, vol. 48, no. 3, pp. 1–26, Feb. 2016.

[12] J. Pohl and A. Noack, ''Automatic wireless protocol reverse engineering,'' in *Proc. 13th USENIX Workshop Offensive Technol.*, 2019, pp. 1–12.

[13] J. Pohl and A. Noack, ''Universal radio hacker: A suite for analyzing and attacking stateful wireless protocols,'' in *Proc. 12th USENIX Workshop Offensive Technol.*, 2018, pp. 1–24.

[14] P. Fraga-Lamas and T. M. Fernández-Caramés, ''Reverse engineering the communications protocol of an RFID public transportation card,'' in *Proc. IEEE Int. Conf. RFID (RFID)*, May 2017, pp. 30–35.

[15] J. Nikander, O. Manninen, and M. Laajalahti, ''Requirements for cybersecurity in agricultural communication networks,'' *Comput. Electron. Agricult.*, vol. 179, Dec. 2020, Art. no. 105776.

[16] M. Freyhof, G. Grispos, S. Pitla, and C. Stolle, ''Towards a cybersecurity testbed for agricultural vehicles and environments,'' 2022, *arXiv:2205.05866*.

[17] A. N. Alahmadi, S. U. Rehman, H. S. Alhazmi, D. G. Glynn, H. Shoaib, and P. Solé, ''Cyber-security threats and side-channel attacks for digital agriculture,'' *Sensors*, vol. 22, no. 9, p. 3520, May 2022.

[18] *GNU Radio Website*. Accessed: Dec. 2022. [Online]. Available: http://www.gnuradio.org

[19] *HackRF One Website*. Accessed: Dec. 2022. [Online]. Available: https://greatscottgadgets.com/hackrf/one/

[20] *BladeRF Website*. Accessed: Dec. 2022. [Online]. Available: https://www.nuand.com/bladerf-2-0-micro/


**ROBERTO CAVIGLIA** (Student Member, IEEE) received the bachelor's and master's degrees in electrical engineering from the University of Genoa, in March 2019 and March 2021, respectively, where he is currently pursuing the Ph.D. degree with the SCNL Laboratory. His master's thesis on cybersecurity threats and attacks detection methods of a storage system in a microgrid. He collaborates with Sababa Security SpA. His research interests include cybersecurity in automotive environment and the network security of industrial control systems.

**GIOVANNI GAGGERO** (Member, IEEE) received the M.Sc. degree in electrical engineering and the Ph.D. degree in electronic and telecommunication engineering from the University of Genoa. He is currently a Postdoctoral Research Fellow with the Satellite Communications and Heterogeneous Networking Laboratory, University of Genoa. His research interests include the network security of industrial control systems, microgrids, and smart grids.

**GIANCARLO PORTOMAURO** received the "Laurea" degree in computer science engineering from the University of Genoa, Italy, in 2002. He is currently a Research Fellow with the Satellite Communications and Networking Laboratory (SCNL), University of Genoa. Since May 2001, he has been with the Italian Consortium of Telecommunications (CNIT), University of Genoa Research Unit, where he was a Research Consultant (Research Staff) with SCNL. His research interests include the emulation of commutation of on board satellite systems, the reliable quality of service network satellite for multimedia applications, software for the satellite emulation systems, wide network systems, installation and training for video conference tools for remote training and instruments access, design, and the realization of event simulator for heterogeneous packet switching networks.

**FABIO PATRONE** (Member, IEEE) is currently an Assistant Professor with the Satellite Communications and Heterogeneous Networking Laboratory, University of Genoa. His research interests include routing, scheduling, and congestion control algorithms in satellite, vehicular, and sensor networks, and the employment of networking technologies, such as network function virtualization and software defined networking for the integration of these networks with the terrestrial infrastructure within 5G.

**MARIO MARCHESE** (Senior Member, IEEE) received the "Laurea" degree (cum laude), in 1992, and the Ph.D. degree in telecommunications from the University of Genoa, in 1997. From 1999 to January 2005, he was with the Italian Consortium of Telecommunications (CNIT), where he was the Head of Research. From February 2005 to January 2016, he was an Associate Professor, and since February 2016, he has been a Full Professor with the University of Genoa where, he has been a Rector's Delegate to Doctoral Studies, since December 2020. He is the author of the book *Quality of Service over Heterogeneous Networks* (John Wiley & Sons, Chichester, 2007), and the author/coauthor of more than 300 scientific works, including international magazines, international conferences, and book chapters. His research interests include networking, the quality of service over heterogeneous networks, software-defined networking, satellite networks, network security, critical infrastructure security, and intrusion detection systems.