A Zero-Trust Methodology for Security of Complex Systems With Machine Learning Components

Conference Paper · August 2021

DOI: 10.1115/DETC2021-70442

4 authors, including:

Douglas Lee Van Bossuyt
Naval Postgraduate School
135 PUBLICATIONS **937** CITATIONS
SEE PROFILE

Nikolaos Papakonstantinou
VTT Technical Research Centre of Finland
81 PUBLICATIONS **1,045** CITATIONS
SEE PROFILE

Bryan O'Halloran
Naval Postgraduate School
64 PUBLICATIONS **476** CITATIONS
SEE PROFILE

Proceedings of the ASME 2021
International Design Engineering Technical Conferences and
Computers and Information in Engineering Conference
IDETC-CIE2021
August 17-19, 2021, Virtual, Online

DETC2021-70442

A ZERO-TRUST METHODOLOGY FOR SECURITY OF COMPLEX SYSTEMS WITH MACHINE LEARNING COMPONENTS

Britta Hale
Department of Computer Science
Naval Postgraduate School
Monterey, California 93943
Email: britta.hale@nps.edu

Douglas L. Van Bossuyt[*]
Department of Systems Engineering
Naval Postgraduate School
Monterey, California 93943
Email: douglas.vanbossuyt@nps.edu

Nikolaos Papakonstantinou
VTT Technical Research Center
Espoo, Finland
Email: Nikolaos.Papakonstantinou@vtt.fi

Bryan O'Halloran
Department of Systems Engineering
Naval Postgraduate School
Monterey, California 93943
Email: bmohallo@nps.edu

ABSTRACT

Fuelled by recent technological advances, Machine Learning (ML) is being introduced to safety and security-critical applications like defence systems, financial systems, and autonomous machines. ML components can be used either for processing input data and/or for decision making. The response time and success rate demands are very high and this means that the deployed training algorithms often produce complex models that are not readable and verifiable by humans (like multi layer neural networks). Due to the complexity of these models, achieving complete testing coverage is in most cases not realistically possible. This raises security threats related to the ML components presenting unpredictable behavior due to malicious manipulation (backdoor attacks). This paper proposes a methodology based on established security principles like Zero-Trust and defence-in-depth to help prevent and mitigate the consequences of security threats including ones emerging from ML-based components. The methodology is demonstrated on a case study of an Unmanned Aerial Vehicle (UAV) with a sophisticated Intelligence, Surveillance, and Reconnaissance (ISR) module.

1 Introduction

With the growing interest in machine learning (ML) applications and solutions, critical systems engineering choices must start incorporating hitherto unaccounted risks from potential threats associated with ML data collection, training, and ML-based decisions. Complex system design and risk assessments are not foreign to system design and risk assessments: cyber-physical systems such as drones, automated factories, autonomous vehicles, financial investment systems, virtual assistants, etc. have all benefited from such analysis [1–3]. Thus, this work attempts to account for and adapting to the risks posed by increasingly automated systems that use advanced ML techniques in a systems engineering context.

Past losses of systems such as the Lockheed Martin RQ-170 Sentinel that crash-landed in an adversary's territory provide the inspiration for this work. In the case of the RQ-170, press reports indicate that the ML aboard may have been fooled into descending until the air frame had a semi-controlled crash-landing. The

[*]Address all correspondence to this author.

culprit may have been spoofed Global Positioning System (GPS) data being intentionally injected into the GPS sensors by the adversary. While laboratory experiments had demonstrated the possibility of hijacking a UAV via GPS spoofing, to our knowledge the RQ-170 incident was the first publicly reported incident possibly caused by this attack on the UAV's ML. In the case of the RQ-170, the air frame made a number of appearances in the adversary's state-run media and likely was analyzed by adversary engineers [4–9].

A string of highly publicized vehicle accidents caused by drivers blindly following GPS navigation directions provides further inspiration. In these situations, a change in roadway configuration, an error in a map database, and other similar issues led a number of drivers to drive their vehicles off of cliffs, into lakes, and into other perilous situations. The drivers generally trusted the ML that developed the directions their GPS devices provided, and did not adequately question the directions for accuracy and safety [10, 11].

The problem of blindly trusting ML extends into high-level autonomous ground vehicles such as self-driving cars. Human reaction speed is sufficiently slow that an issue with the ML such as GPS spoofing, phantom images, and other physical-world attacks on ML classification that a human taking control of a ML piloted vehicle may not occur before the calamity occurs. In the case of a ML-piloted system that has minimal or no human oversight (e.g.: a UAV operating in a denied environment where RF and optical communications are unavailable or unreliable), recovery may be impossible [11–13].

While much effort goes into anti-spoofing and anti-hijacking technologies, and ML error correction, events continue to happen. Adversaries who may wish to destroy or capture a ML-equipped system continually improve techniques for carrying out such activities. Further exacerbating the situation is increasingly complex and opaque ML that is not deterministic in behavior. In other words, in many cases ML cannot be 100% verified in its responses to situations it may encounter.

One might question why ML is being used in safety-critical and defense domains when such drawbacks exist. The new capabilities that ML enables in modern systems have been deemed worth the potential risks. In many cases, ML can do a job with fewer errors than a human and no need for rest periods. However, the latest advances in ML and relatively few reported direct adversary actions against ML-equipped systems we believe has produced a level of complicity in systems engineering development processes that we intend this paper to begin to address.

Similar challenges to those facing systems with ML are already being addressed in the safety engineering domain for different types of critical systems that need to achieve a minimum safety risk. System components are never absolutely reliable – in other words no system component can be trusted and it is assumed that a component will fail sooner or later. Apart from increasing the reliability at a component level, such as by using new technologies, special system configurations and multiple layers of defence are deployed to decrease the overall risk to an acceptable level. For instance, Programmable Logic Devices (PLDs) are not designed for safety-critical applications (although manufacturers often claim otherwise) yet are still used in applications such as missile systems. When a PLD is initially powered on, there exists a brief moment where the device is capable of producing spurious output signals. In spite of this shortcoming, PLDs are used extensively in missile systems in safety-critical roles such as to activate and/or ignite rocket motors, separate stages (e.g.: boosters, glide vehicles, etc.), activate wings and control surfaces, detonate warheads, and actuate flight termination systems. While missile systems knowingly uses non-safety-critical hardware for safety-critical applications, relatively few incidents have resulted from their use. Part of the success of PLD use in safety-critical systems is that systems engineers design with the knowledge that PLDs cannot always be trusted.

In summary: a challenge exists with the increasing uptake of ML in systems, and with increased ML sophistication and opacity. **The contribution of this paper** is a system design methodology based on Zero-Trust and defence-in-depth principles for increasing the level of security and assurance when ML is incorporated to the system under development.

2 Background

2.1 Complex systems design challenges

Typically complex systems are developed through a system design process that attempts to follow a model such as the V-model, the waterfall model, the spiral model, etc. of how the design process should proceed [14]. Complex systems are generally too complex for any one person to fully understand all aspects of the systems and all potential outcomes for all potential inputs. Thus, many subject matter experts (SMEs) are employed to undertake specific portions of a system design. Often times, systems engineers are used as the "glue" that binds together different SME efforts. In many cases, systems engineers also conduct analyses to identify potential issues that cross discipline boundaries (e.g.: safety, security, and failure analysis).

Numerous methods such as Failure Modes Effects and Criticality Analysis (FMECA), Probabilistic Risk Assessment (PRA), hazard analysis, reliability analysis, and others exist to identify potential issues that cross discipline boundaries and can have adverse impact on the system [1–3, 14–17]. However, systems still suffer "black swan events" where a deleterious emergent system behavior occurs that was either not predicted or ignored during system design [3, 18, 19]. Research is ongoing in this area in an attempt to address these issues and produce safer, more reliable systems.

2.2 Current standards for security of critical infrastructure

Generic and domain specific standardization efforts help with security related challenges when developing systems. NIST guidelines cover the management of information security risks [20] while the IEC 62443 series on security of industrial networks and communication systems [21] focuses on management and prevention of security risks and introduces key security concepts like security levels, defense-in-depth, and segregating systems into zones. A three part ISO standard, ISO/IEC 15408 [22], defines a set of security function design requirements and also deals with requirements for security evaluation and assurance. The ISO/IEC 27000 standard series (27001,2,3,4,5) [23] focuses on cybersecurity technologies, as well as confidentiality and privacy. These standards can help to inform complex system design processes with a cyber component such as ML.

2.3 Security of ML-based components, ML verification research, Adversarial ML

Security considerations are critical to ML from the point of data collection and storage for use in training, through training and final model protection. Model training in ML requires a significant amount of data – an aspect that has given rise to legal concerns regarding the protection of data [24]. In some cases, models can be trained without transparent data access, using techniques known as privacy preserving protocols [25,26]. In all cases, protecting and tracing data from the point of collection, through aggregation, and to training is essential to prevent data tainting. Through data tainting or injection, an adversary may not only affect the performance of the model in general but may also cause unexpected model behavior. This latter issue is known generally as *adversarial AI* or *backdoors*.

Adversarial AI can take many forms [27]. It usually involves some tainted training data that contains a "trigger"; when the model is later used on data that contains the trigger, it may falsely classify the data. There has also been work on developing *triggerless* adversarial AI for deep neural networks, wherefore the adversary does not actually need to taint the training data set [28]. While this variant of adversarial AI requires several conditions to be met, such as the network type, it also indicates progress towards making such backdoors in a model even more difficult to control for or detect.

Methods for assessing the validity ML algorithms include the training of two separate models, where one provides a prediction rationale for the second [29]. While this method makes headway towards verifying the ML model, it is not guaranteed to protect against adversarial AI methods – including those yet to be developed. Some methods of hardening an ML model against adversaries, such as *adversarial training*, have the potential to actually introduce backdoors unintentionally [30].

In yet another variation of validating ML models, explainable AI (XAI) [31–34] aims to provide verification of the model output through an interpretation that is accessible to users. Such approaches may simply employ an human-accessible interface to the results or may also rely on a second model for prediction rationale. Essentially, explainable AI aims for a human-in-the-loop approach towards building verification and, as a direct consequence, trust in the system.

While the above methods improve the evaluation of ML, simple validation is an incomplete solution towards nullifying full system security risk. ML models may degrade over time [35], and some types of ML such as Federate Learning can lead to cross-over bleeding between models, leading to more complex adversarial attack strategies [36]. All of this points to the criticality of strategic system risk assessment and non-trivality of risk assessment for systems with ML components.

2.4 Zero-Trust security paradigm

Security assessment is dependent on a threat model, which is built to capture specific adversarial behavior and define aspects that are in scope. Within security analyses this naturally poses a challenge, as an analysis can indicate security within the threat model while the system is still vulnerable to attacks and adversarial behavior not specifically described. From a system engineering view, analyses focusing on core component interaction may therefore yield positive results, even if a trivial error on the part of human interaction with those components would render the entire system insecure. This poses a risk to the overall system. Such challenges have lead to research increasingly aiming to capture risk assessments in more complex systems [37–39].

To address these challenges, the concept of a Zero-Trust architecture has been used as a risk assessment framework [40]. As a premise, Zero-Trust assumes that any component in the system or outside the system could be faulty or compromised (i.e. capable of introducing accidental or adversarial effects). Zero-Trust aims to capture system components, interaction, and human users. No item within the system bounds is out-of-scope. Extensive work has looked at applying the Zero-Trust framework to areas such as IoT, Big Data, and Infrastructure as a Service [41–44]. Indeed, the Zero-Trust use has expanded in adoption as far as standardization by U.S National Institute of Standards and Technology (NIST) [45].

As ML use develops and enters normal system use, so must assessments expand to consider risks introduced in the data-for-training acquisition, model training, and application of ML [46]. The use of ML for adversarial purposes points to the fact that, while ML usefulness demands its integration, system security assessors cannot afford to ignore potential risks introduced by it. In short, just as human error and adversarial capability entered system risk assessment, so must the ML. Zero-Trust provides a grounding assumption for accepting and balancing the potential risks of ML use.

2.5 Zero-Trust in Systems Engineering

The Zero-Trust paradigm has recently been introduced into the systems engineering community through treating both hardware and humans as potential threats to a system regardless of provenance. Hybrid attack-fault trees have been developed to integrate failure analysis and security threats. At their core, the methods being developed for systems engineers that use the Zero-Trust paradigm trust no one and no hardware involved in the design, manufacture, operation, and maintenance of systems. However, these methods have largely ignored the role that ML can play in introducing new avenues of attack. While safety interlocks to prevent PLDs from triggering accidental detonation of warheads have been standard for decades, similar techniques have not yet been applied to ML when implemented into many safety-critical and defense applications. Instead, more often than not, the ML is implicitly trusted to do its job and not be compromised.

3 Methodology

In this section we introduce a methodology for system security based on Zero-Trust and defense-in-depth and provide a simple example. The goal of the proposed methodology is to help the design of systems with security and safety measures that can prevent and mitigate threats posed by ML components, even if the specific attack vector is unknown at the time of the system development. All system components (humans, engineered systems, ML, and external environment) are considered as untrusted and thus a potential starting point of an attack. This is even more important for ML components with decision-making authority or that can feed into decision-making when the ML components have non-verifiable behavior. The needed security and safety measures can be based on knowledge databases with generic solutions given the challenge, like e.g. to develop two separate ML components by isolated teams and with voter block logic. The proposed methodology follows the workflow shown in Figure 1 and it sources information from all the system life-cycle phases (design, testing, deployment, operation, maintenance, decommissioning, etc.).

Step 0: Before the main steps of the methodology are entered, safety and security requirements from a zero-trust and defense-in-depth perspective for the system must be set. We advocate for setting a specific number of defense-in-depth elements required of key components to the successful operation of the system. Further, we recommend a requirement that specifies how diverse control systems (digital, analog, human, etc.) must be. Finally, we suggest a requirement be set that guides identification of internal and external threats. These pre-sets represent the acceptable complexity bounds and security priorities.

Step 1: In the first step the goal is to identify the life-cycle phases in which the system under study contains specific system elements that hold sensitive information or elements that are able to cause harm. For instance, the maintenance phase of a system's life-cycle could open a ML component being upgraded to vulnerabilities from contractor maintenance personnel knowingly or unknowingly introducing adversarial AI.

Step 2: The next step calls for the identification of the specific system elements for every life-cycle phase that are "critical" (responsible for holding critical information or can directly cause harm) and the creation of a dependency model of the system and its environment. In other words, identify the potential components or subsystems that have a significant consequence associated with their loss, destruction, or disablement, and then develop dependency trees for said components. For example, a subsystem that if operated incorrectly by a hacked ML component could destroy itself [47] should be identified in this step.

Step 3: In Step 3, the aim is to identify the signal, material, and energy interfaces (we recommend the Functional Basis for Engineering Design (FBED) ontology [48] for this purpose) of the critical system elements within the system under study and with the natural, constructed environment as well as with other external systems. Furthermore the influences on the critical system elements are analyzed for all of the relevant life-cycle phases (e.g. design, component supply, assembly, testing, deployment, and maintenance).

Step 4.N: This step provides additional defence-in-depth layers. It calls for all the elements that interface or influence a critical system element to be analysed in the same way as a critical system element. In this concept, if only the interfaces/influences of the critical system elements are considered, then they have only one layer of protection; on the other hand, if we also consider the interfaces/influences on all the elements that are linked to the critical system element, then we add a defence-in-depth layer, etc. The additional levels of defense may not deploy the full set of controls, depending on the mission of the system under study and the resources available.

The following steps 5 and 6 take into account the Zero-Trust principle. Every interface and influence has the potential to compromise security. Authentication, monitoring, and security checks need to be deployed in any case.

Step 5: The fifth step focuses on the internal and external interfaces that associate with every critical system element where an internal or external threat could manifest. Primary and diverse secondary controls should be deployed for authentication, monitoring, for blocking data leaks and for interlocking actions that can cause harm.

Step 6: The influences relevant to the critical system elements are considered in Step 6. This step calls for the adoption of controls (primary and diverse secondary) for authentication, monitoring and inspection for every interaction of an influencing element with the critical system element.

Decision Point: Now the results of the methodology to this point are compared against the requirements set in Step 0. If the

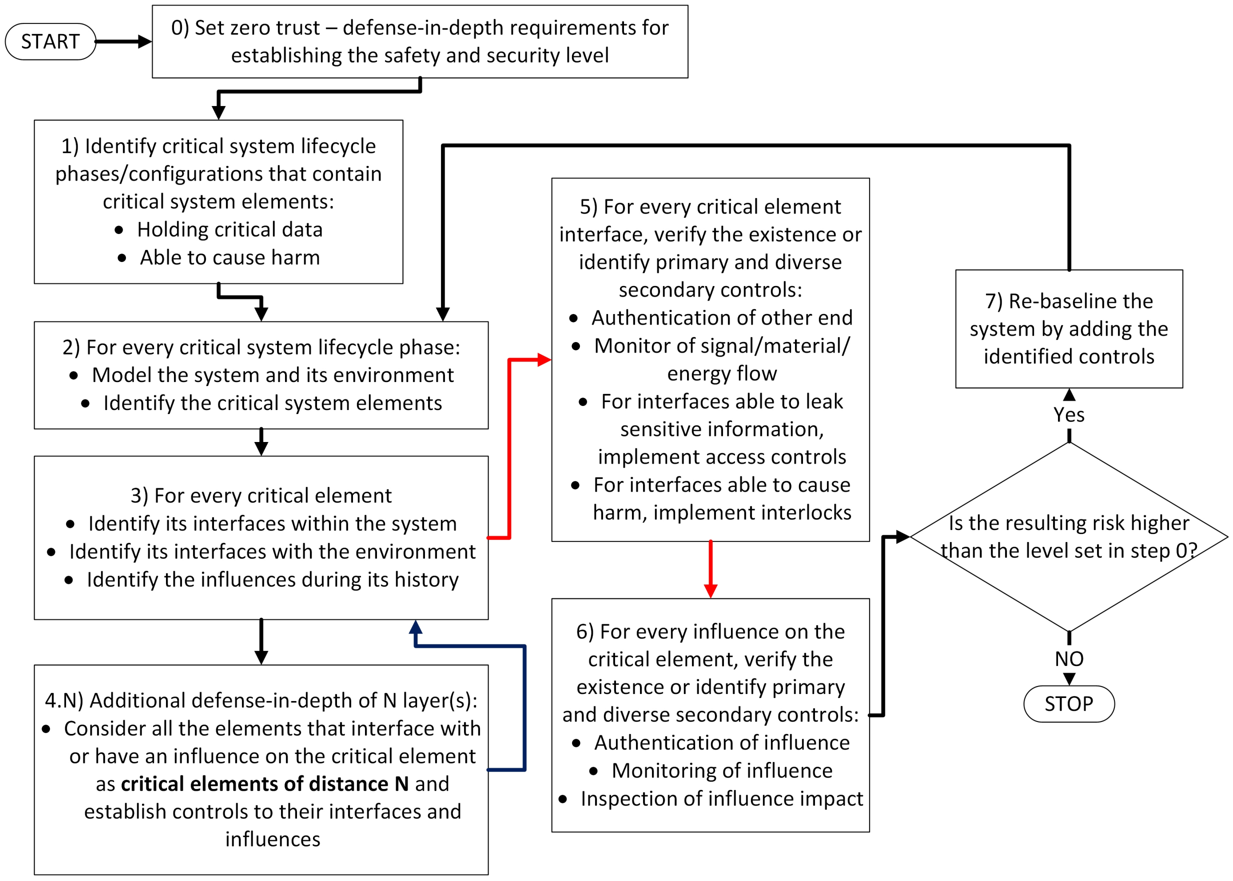
START
0) Set zero trust – defense-in-depth requirements for establishing the safety and security level
1) Identify critical system lifecycle phases/configurations that contain critical system elements:
• Holding critical data
• Able to cause harm
2) For every critical system lifecycle phase:
• Model the system and its environment
• Identify the critical system elements
3) For every critical element
• Identify its interfaces within the system
• Identify its interfaces with the environment
• Identify the influences during its history
4.N) Additional defense-in-depth of N layer(s):
• Consider all the elements that interface with or have an influence on the critical element as critical elements of distance N and establish controls to their interfaces and influences
5) For every critical element interface, verify the existence or identify primary and diverse secondary controls:
• Authentication of other end
• Monitor of signal/material/ energy flow
• For interfaces able to leak sensitive information, implement access controls
• For interfaces able to cause harm, implement interlocks
6) For every influence on the critical element, verify the existence or identify primary and diverse secondary controls:
• Authentication of influence
• Monitoring of influence
• Inspection of influence impact
Is the resulting risk higher than the level set in step 0?
Yes
NO
7) Re-baseline the system by adding the identified controls
STOP

FIGURE 1. Methodology Process

requirements are met by the existing system, then the methodology is complete. If the requirements are not met, then the methodology proceeds to Step 7.

Step 7: In Step 7, the system design is re-baselined based upon the design changes proposed in Steps 4-6. This may require subject matter expert involvement to develop sufficient models and architectures for use in the next step. After this step, the methodology loops back to Step 2 for re-analysis to confirm that the system meets the zero-trust requirements.

3.1 Simple example

The simple example shown in Figure 2 can help to demonstrate the basic flow of the proposed methodology. The first step of the methodology is to identify in which life-cycle phases the system in question either contains valuable information or is capable of causing harm. For this system it identifies that during operation this is true, so the operation life-cycle phase is deemed to be critical.

The system has a requirement of a primary and a redundant set of controls, authentication methods, monitoring, and inspection. In the operation phase, the system has a configuration that includes 5 components and the dependencies shown in Figure 2. Step 2 of the methodology ask for the identification of the critical system elements. The System element C is considered critical because it holds sensitive information and the element E is critical because it can potentially cause harm.

The internal interfaces as well as the interfaces to the environment of element C and E are identified in Step 3, as well as their past influences (e.g. during design, supply, deployment, maintenance, etc.). For example for the element E interfaces to element D and to external environment are identified. Also one actor is identified as an influence during design, two actors during supply and assembly and three during testing, deployment and maintenance. All these interfaces and influences are collected to a list to be used in Step 5 (interfaces) and step 6 (influences).

For additional *N* layers of defence-in-depth, system or environment elements connected to the critical elements are analysed in Step 4.N and their interfaces and influences are added to the lists for Step 5 and 6. In this example the interfaces/influences of element D, of the environment elements and the influences of

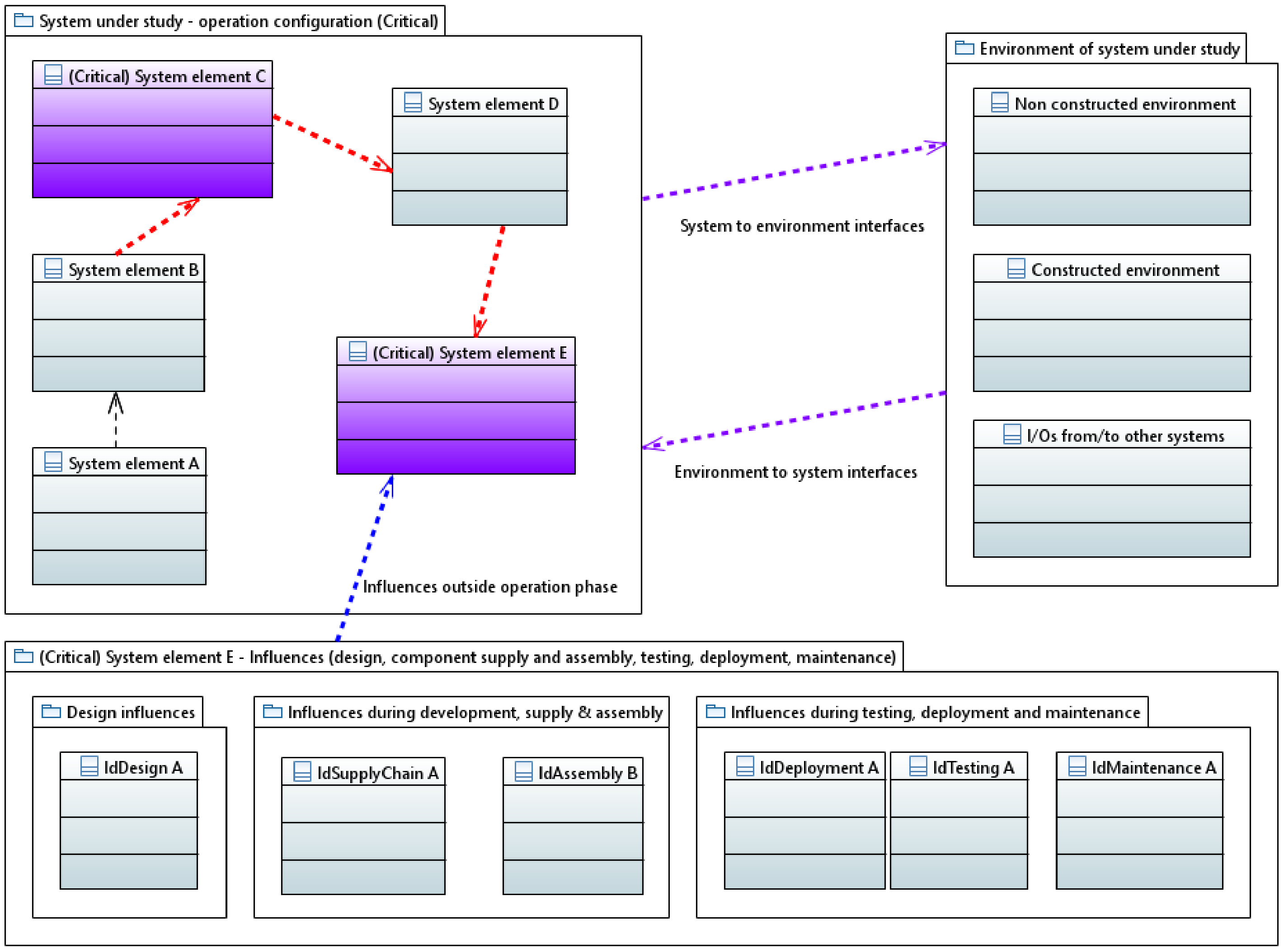
System under study - operation configuration (Critical)
(Critical) System element C
System element D
System element B
(Critical) System element E
System element A
Influences outside operation phase
Environment of system under study
Non constructed environment
Constructed environment
I/Os from/to other systems
System to environment interfaces
Environment to system interfaces
(Critical) System element E - Influences (design, component supply and assembly, testing, deployment, maintenance)
Design influences
IdDesign A
Influences during development, supply & assembly
IdSupplyChain A
IdAssembly B
Influences during testing, deployment and maintenance
IdDeployment A
IdTesting A
IdMaintenance A

FIGURE 2. Simple Example Model

element E (IdDesignA, IdSupplyChainA, etc) are also added to the lists.

When the lists of critical interfaces and influences is compiled, then in Step 5 a primary and a secondary control is established to ensure authentication, monitoring and access control/interlock. E.g. for a communication interface between an operator and a drone, there needs to be a primary and a diverse secondary technology for authenticating the operator to the drone, ways to monitor this communication and flag suspicious behavior and ways to block commands that can cause harm (e.g. the need to have the "go ahead" from a second operator and a supervisor officer).

In Step 6 a similar process of establishing controls is focusing on the influences to the critical elements. For element E 6 different actors (humans, ML tools, etc) have been identified to have an influence during different life-cycle phases. The goal is now to establish primary and diverse secondary methods for authentication, monitoring and inspection of the impact of the influence. As an example, if the element E includes software that has been (partially) tested by a test engineer, IdTesting, then there need to be ways to authenticate the engineer, to monitor her interaction with the component and lastly to inspect the component after the testing is done to confirm that it has not been tampered with.

Now the system is evaluated to determine if it meets requirements or needs to be rebaselined and reanalyzed. Step 7 is activated if the system risk is not acceptable. In this case, the identified controls from Steps 5 and 6 are included and the system re-baselined. The system is then re-evaluated.

For the interfaces/influences identified in Step 4.N (additional layers of defence), depending of the critically of the mission of the system and the available resources, it may be justified that not all controls are implemented.

An objective of this approach is resilience of systems containing ML components for decision support and data processing. If in this simple example the system element E did contain ML technology, the proposed methodology can guide the application of controls to prevent and mitigate security threats. Part of the effort is placed during the influences on the ML components but also effort is put in controlling the interfaces and interlocking the actions that can cause harm.

4 Case Study

Consider an UAV with a sophisticated intelligence, surveillance, and reconnaissance (ISR) module aboard flying a patrol route through a mountainous area with deep valleys where remote control of the system is via radio frequencies (RF). The RF transmission can become unreliable due to mountains blocking RF signals, weather and other natural interference issues, and active interference generated by an adversary. To prepare for this, the UAV is designed to fly autonomously through areas where RF communications are unreliable using an ML component for autonomous control. Furthermore, the ISR package aboard is equipped with a self-destruct system to eliminate sensitive components in the event of flight failure to ensure that such components do not fall into adversarial hands. While under autonomous control, an ML-controlled decision process aboard has authority to activate the ISR self-destruct system if it detects one or more of several events listed in Table 1 occurs. The specific events that allow the ML to activate the ISR self-destruct system are derived from past real-world experience and lab demonstrations [8, 49–52].

TABLE 1. ISR self-destruct system events.

1	UAV is in eminent danger of crashing
2	UAV RF control has been commandeered by adversary
3	UAV flight sensors (e.g.: GPS, LIDAR, etc.) are receiving compromised data

The UAV together with the constituent ISR package form the overall UAV system. The UAV system was designed, integrated, and manufactured by a number of contractors and subcontractors. Each subsystem within the UAV is designed and manufactured by a different subcontractor, and components within each subsystem may further be designed and/or manufactured by different subcontractors; thus supply-chain trust is widely distributed. Subsystems and components may be physical hardware or software. The ML model onboard the UAV system is trained and tested in a virtual environment prior to deployment.

TABLE 2. Potential Adversary Scenarios

1	Adversary attempts to destroy UAV system
2	Adversary attempts to capture ISR package
3	Adversary attempts to destroy ISR package
4	Adversary attempts to commandeer UAV system
5	Adversary attempts to evade detection by ISR package
6	Adversary attempts to confuse UAV operators at FOB
7	Adversary attempts to provoke a response from the FOB by posing as an imminent threat

In addition to the system design considerations we also consider maintenance during use. This applies to both hardware and software. We assume that the ML model onboard the UAV system continues to be trained during operation using successful mission data. Adversaries may be present in the area of operation and therefore may affect the training data (e.g. by affecting sensor or traffic data, the adversary can mimic an attack of the form in Table 2), thus hardening the model against eventual, subtle real attacks. Furthermore, maintenance covers part replacements that may be provided or handled by various contractors or subcontractors. During the repair process, such actors may also gain access to the ML model onboard.

Figure 3 graphically shows a typical patrol route and UAV system mission. The UAV system departs from and returns to a forward operating base (FOB). The UAV system patrols along several mountain valleys with the primary goal of detecting adversary activity in the area using the ISR package.

While an adversary's ultimate goal as well as attack method may be unpredictable – even highly adaptable – several potential scenarios have been postulated including those listed in Table 2. Potential adversary actions have also been postulated as shown in Table 3.

In the case study, we assume that several potential strategies. Table 4 are developed and implemented to determine when to activate the ISR self-destruct system based on the events outlined in Table 4. The ML-controlled decision process determines when conditions are met based on the ISR self-destruct strategies. There are several potential outcomes of the ISR self-destruct strategies outlined in Table 5. Replacing the ISR package in the event of an unnecessary ISR self-destruct event is expensive and time-consuming, and the FOB commander has a strong interest in ensuring the ISR package only is destroyed when necessary.

Figure 4 shows a high-level system model of the UAV sys-

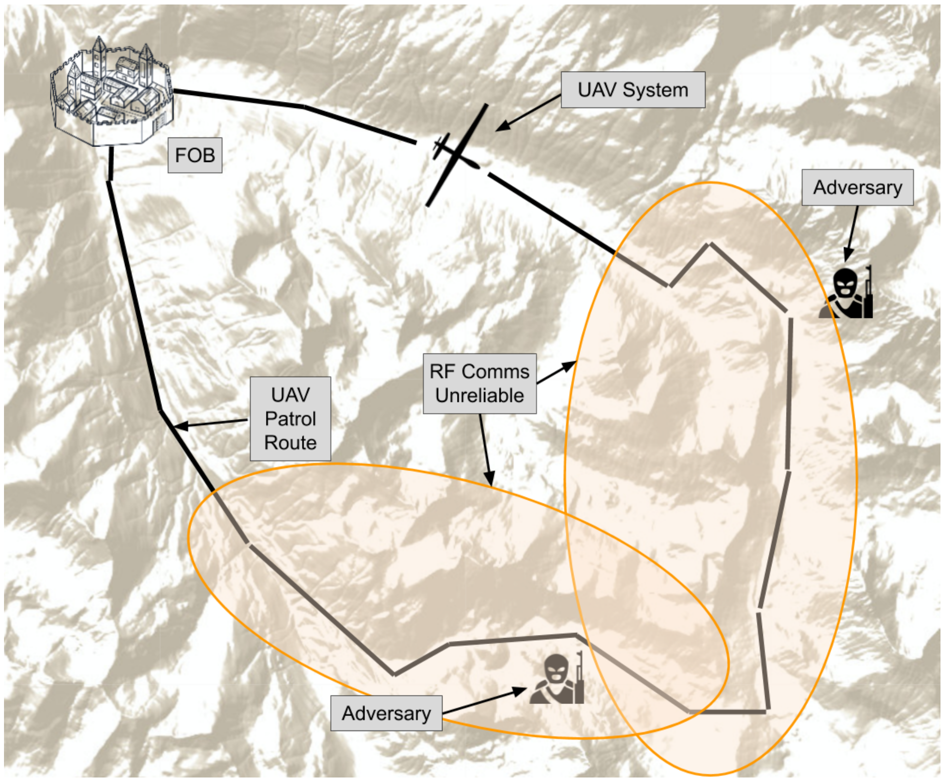

FIGURE 3. Depiction of typical patrol route and mission of UAV system.

TABLE 3. Potential Adversary Actions

1	Adversary spoofs GPS signal
2	Adversary spoofs/hijacks RF comms
3	Adversary conditions ML to respond in a specific way in order to lure ML into flying into a box canyon
4	Adversary interferes with LIDAR sensor
5	Adversary employs anti-air defences (e.g.: flak, missile, etc.)
6	Adversary presents multiple targets for ISR package to identify

TABLE 4. ISR Self-Destruct Considerations

1	Deviation from planned course
2	Loss of altitude unplanned within autonomous control
3	Unusual C2 traffic activity
4	Unusual sensor activity (LIDAR, GPS, etc.)
5	Unusual propulsion or energy
6	Operator override

tem. Figure 5 shows a system hierarchy diagram decomposed into subsystems and components. Each component has an associated fault tree as shown. Each fault tree contains failure information on design, manufacturing, and operation from a Zero-Trust perspective. The Zero-Trust fault tree graphically demonstrates how each major phase of the system engineering process (e.g.: conceptual design, component design, integration, manufacturing, verification and validation, operation, maintenance, upgrade and overhaul, etc.) can be a potential source of a fault which may cause the system to fail in operation.

For this case study, the methodology workflow starts by developing zero-trust requirements (Step 0). The requirement is set as one layer of defense in depth. Next, Step 1 is completed where the system life-cycle phases where there is possibility for loss of critical information/data or for causing harm are identified. It is identified that during the mission execution phase where during autonomous flight there is possibility to leak information/knowledge from the ISR module.

The next step (2) is to model the UAV system in its mission operation configuration, and focus on the internal and ex-

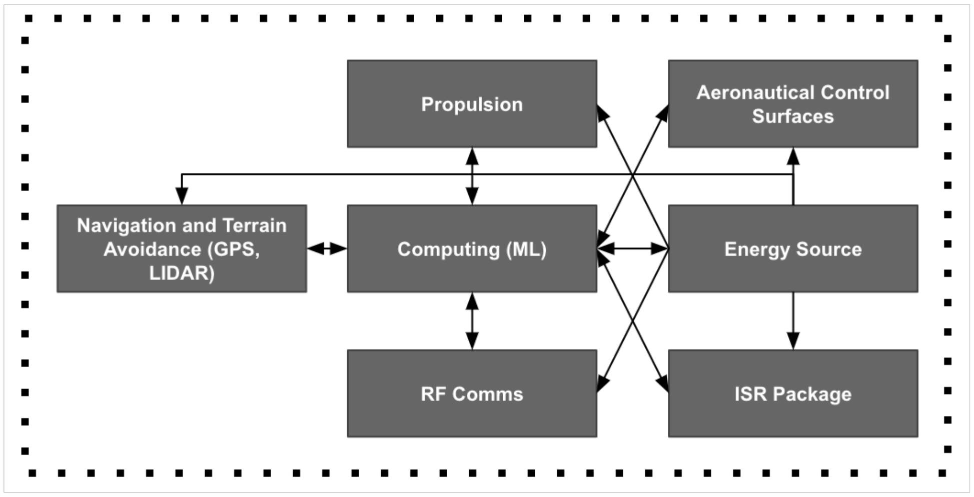
Propulsion
Aeronautical Control Surfaces
Navigation and Terrain Avoidance (GPS, LIDAR)
Computing (ML)
Energy Source
RF Comms
ISR Package

FIGURE 4. High-Level System Model of UAV System.

TABLE 5. ISR self-destruct outcomes.

Diag.	Action	Outcome
Corr.	Destroy ISR package	ISR package destroyed
	Retain ISR package	ISR package retained
Incorr.	Retain ISR package	ISR package destroyed
	Destroy ISR package	ISR package retained

ternal (environment) interfaces of the ISR system element (see Figure 6). As described above, the ISR heavily relies to the security of the drone itself (it operates in a hostile environment) as well as signal data that it receives (sensor data as well as RF communication data flows). The ISR is a mechatronic component that has been influenced by many actors during design/development/testing/deployment/maintenance etc. In Step 3 these interfaces and influences are identified and lists are prepared for Steps 5 and 6. Figure 6 shows some examples at a high level, but in a real application this list would be more specific and detailed.

In this case study, one layer of defence was deemed to be enough. However, if needed more layers can be added by further analyzing the environment, internal system components and actors relevant to the ISR, as described in Step 4.N (at N levels of depth, depending on the additional defence layers desired). I.e. here we focus only on the ISR module and its interfaces/influences, but for more layers of defense we could also investigate interfaces and influences of the components linked to the ISR, like the power supply, frame, sensors, the systems the ISR communicates with and the actors that influence the ISR.

Step 5 iterates through the internal and external interfaces of the ISR and establishes primary and secondary diverse measures that ensure authentication, monitoring, and access control to the internal information of the ISR. As an example in this case study, given the hostile environment, many interfaces to the environment were identified as potential carriers of security threats. Namely, the GPS input data, the RF communications, bad/compromised sensor inputs and physical attacks (air defence actions). To protect against leaking the information (data as well as IP property) part of the ISR, monitoring controls were proposed to identify significant disruptions to data flow to the ISR and the flight of the UAV. If the primary and secondary system agree that there is a disruption that might lead to the drone being captured, then they initiate two processes to destroy the ISR information, first a secure erase that wipes collected data and firmware and then a physical destruction sequence that destroys the ISR. Without both the primary and secondary systems agreeing, neither process takes place.

In Step 6 the influences to the ISR during its lifecycle are evaluated and measures are deployed to authenticate, monitor and inspect all interactions between the ISR module and different actors involved to its life. The Zero-Trust principle mandates that everyone who interacted with the ISR model actively (e.g. development, testing, maintenance) or passively (e.g. transportation) need to be assessed and controls need to be in place.

In Steps 5 and 6 there was an assessment of the redundancy and diversity controls that are handling potential threats emerg-

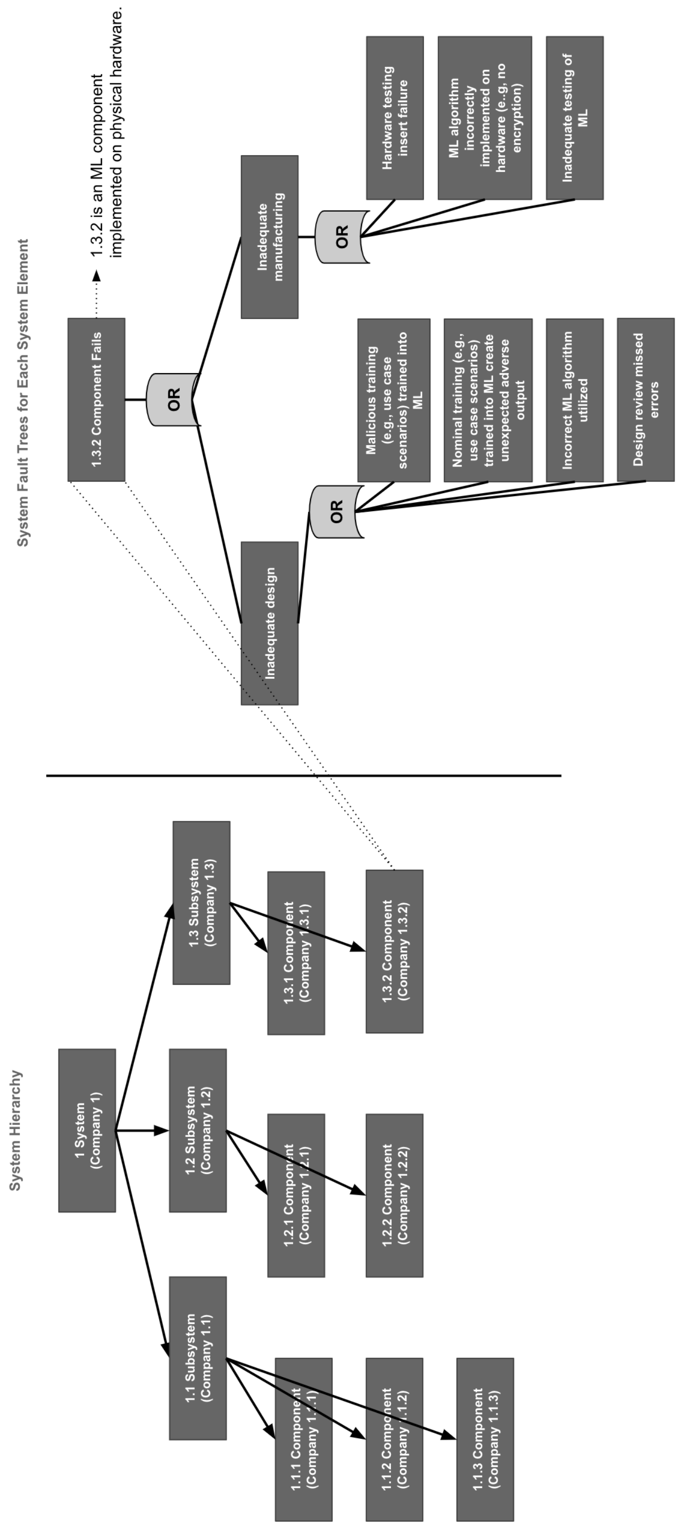
System Hierarchy
1 System (Company 1)
1.1 Subsystem (Company 1.1)
1.2 Subsystem (Company 1.2)
1.3 Subsystem (Company 1.3)
1.1.1 Component (Company 1.1.1)
1.1.2 Component (Company 1.1.2)
1.1.3 Component (Company 1.1.3)
1.2.1 Component (Company 1.2.1)
1.2.2 Component (Company 1.2.2)
1.3.1 Component (Company 1.3.1)
1.3.2 Component (Company 1.3.2)
System Fault Trees for Each System Element
1.3.2 Component Fails
1.3.2 is an ML component implemented on physical hardware.
OR
Inadequate design
Inadequate manufacturing
OR
Malicious training (e.g., use case scenarios) trained into ML
Nominal training (e.g., use case scenarios) trained into ML create unexpected adverse output
Incorrect ML algorithm utilized
Design review missed errors
OR
Hardware testing insert failure
ML algorithm incorrectly implemented on hardware (e.g. no encryption)
Inadequate testing of ML

FIGURE 5. System hierarchy diagram with associated example Zero-Trust fault tree.

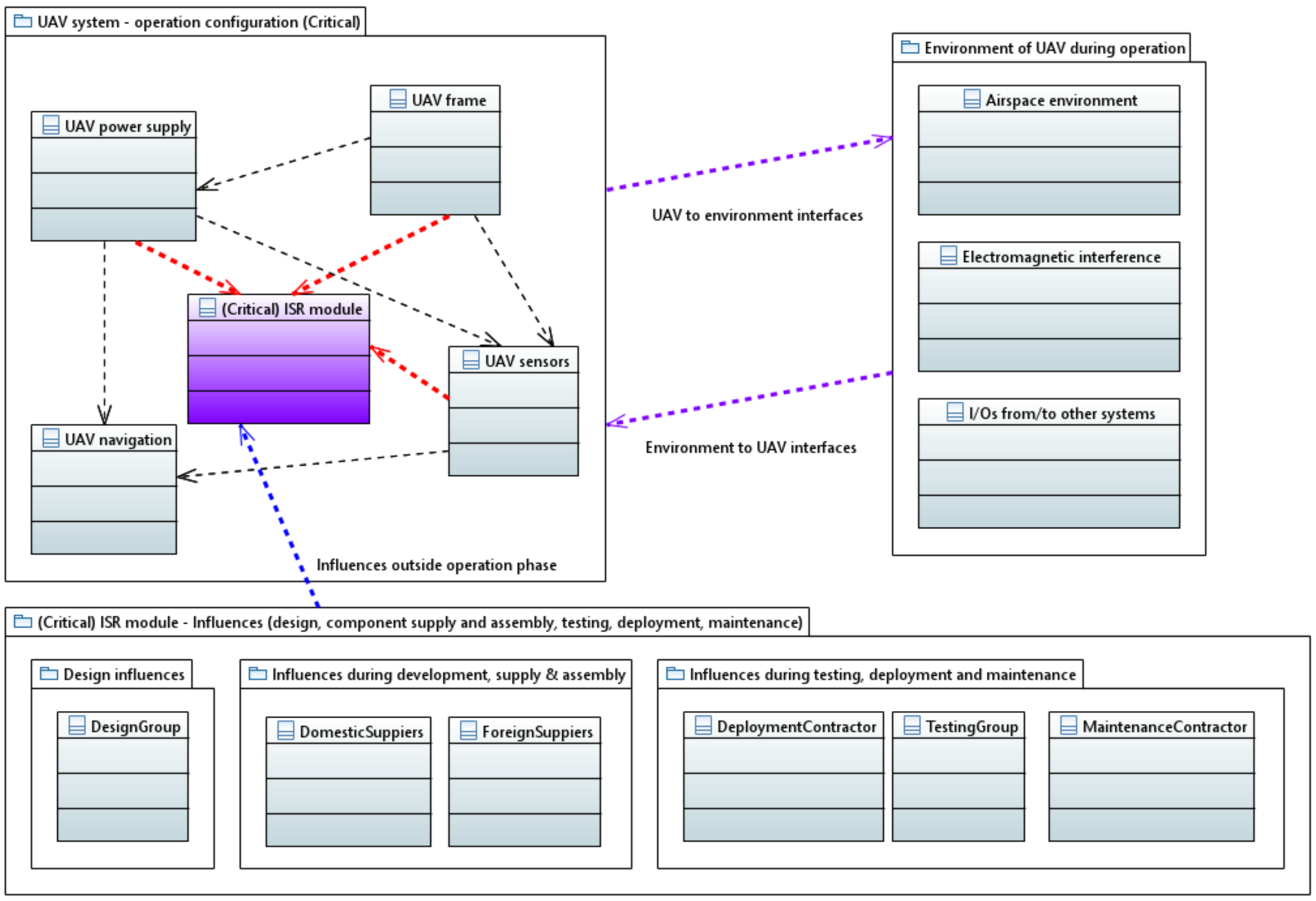
UAV system - operation configuration (Critical)
UAV frame
UAV power supply
(Critical) ISR module
UAV sensors
UAV navigation
Influences outside operation phase
UAV to environment interfaces
Environment to UAV interfaces
Environment of UAV during operation
Airspace environment
Electromagnetic interference
I/Os from/to other systems
(Critical) ISR module - Influences (design, component supply and assembly, testing, deployment, maintenance)
Design influences
DesignGroup
Influences during development, supply & assembly
DomesticSuppiers
ForeignSuppiers
Influences during testing, deployment and maintenance
DeploymentContractor
TestingGroup
MaintenanceContractor

FIGURE 6. Zero-Trust UAV Model

ing from interfaces and influences of the ISR. Now the decision point is reached in the method where the system is evaluated against the requirements set in Step 0. If the controls present were inadequate and new controls needed to be adopted, Step 7 would rebaseline the system with added controls and then the methodology would return to Step 2. If there was no need for redesign, then the workflow ends.

In case of budget (monetary, volume, weight) limitations, the allocated resources for these security mitigation measures and controls can be adjusted towards the more risky interactions, but the Zero-Trust principle mandates that no interaction can be trusted completely.

5 Discussion

Emerging complex systems contain advanced software components, including ML applications like data processing and decision making. This research addresses the challenge of ensuring safety and security of such complex systems despite the paradigm change in critical components, and opens the door for further system design analysis and testing. The proposed methodology utilized established principles of Zero-Trust, defence-in-depth, redundancy, and diversity to guide the practitioner in the analysis of the system under study. It further deploys controls capable of mitigating or preventing loss of sensitive information and other damaging adversarial actions.

A key objective of the methodology is advance identification of potential critical event paths emerging from a specific state of the system or from the influences of its past, and therefore highlight them for preemptive action. This contrasts with the often-used practice of designing prevention and control measures after major incidents happen. The Zero-Trust principle removes the need to rely of the knowledge of pre-existing vulnerabilities or security shortcomings; every component is considered potentially vulnerable and able to propagate attacks.

5.1 Redundancy

Using this method in a system design process may allow for more reliable and trustworthy systems to be rapidly deployed. This is possible because the method helps to explicitly identify potential issues throughout the life-cycle of a system related to ML in a Zero-Trust paradigm. However, the method may add expense and delay to a system design process in certain circumstances, and even new security considerations. For example, in the context of a critical ML-based component, a second, independent ML model may be trained as a redundancy check to reduce security risk to the overall system. The second ML model is in turn a critical component that requires security throughout the lifespan of the model (data aggregation through training and use). Naturally the practitioner should apply common sense in the application of redundancy, since additional layers can significantly increase the number of interfaces and influences that need to be secured. A prioritization may need to be performed, depending on the impact of the security risk (data loss, system damage, or mission compromise) and the available resources.

5.2 Human Element

The human element adds a layer of complexity to system risk assessments, both from a potential adversarial perspective as well as from risk of error. However, they can also provide redundancy through a supervisory defense-in-depth approach. As with every other component under a Zero-Trust approach, there is a trade-off between viewing the human as a point of robustness in the system and a point of failure.

One of the most notable aspects of comparison for ML system components and human system components arises from the concept of artificial intelligence as a term – namely the possibility of replacing the human user entirely with the ML component. However, even as humans are fallible, ML also introduces a failure possibility; accuracy levels returned with an ML prediction can be viewed through the lens of inaccuracy tolerance levels in the same manner as human error. Also, as noted before with adversarial ML, both humans and ML models are potentially adversarial, even if unintentionally. The human/ML correspondence is not one-to-one but rather dependent on the goals of the system and threat model. When optimal security for the full system is desired, using a layered human/ML approach provides more potential defense-in-depth against individual weaknesses, and could be considered as a redundancy check in a similar vein to training two separate ML models.

XAI approaches [31–34] not only have the potential to improve trust in the system by developing human understanding of AI/ML outputs but also, as a consequence, improve the success of using humans in a supervisory role. With an increased understanding of the system, the human is better equipped to provide redundancy so that the ML as a human-on-the-loop regarding decisions.

6 Future Work

This research does not cover the development of specific security controls, monitoring, or authentication techniques. Specific engineering domains have established redundancy solutions that new technologies are required to provide, e.g. diversity or redundancy to a well known security control. Concrete potential extensions and improvements under this paradigm include the following:

1. Future research may improve understanding of prioritization for the interfaces and influences of the critical system elements, in order to keep a balance between security, mission performance and resources. Potential methodology steps can be defined to identify preventive measures or controls that can reliably cover multiple critical components simultaneously. Similarly, it steps can be defined to support identification of a component open to compromise from multiple interfaces/influences.
2. Our defense-in-depth approach can be further refined with expanded work in XAI. As user understanding of specific ML results within various use-cases increases, so does the value of a human-on-the-loop. This in turn brings an overall reduction of risk for systems supporting user input.
3. In a system of systems (SoS) context, the potential diversity of ML across SoS that are cooperating to complete the same mission may introduce new vulnerabilities to the SoS not present with individual systems. A spurious system emission approach such as proposed in [3,18] may be one avenue to address the issue of SoS with diverse ML. However, further research specifically addressing the implications of SoS with diverse ML should be pursued.

7 Conclusion

The proposed methodology is a first step towards the formalized evaluation of mission-critical systems focusing on interfaces and past influences based on the principle of Zero-Trust and defence in depth. This is a lengthy process and more research is required to help the assessor and the designer to re-use past knowledge and exploit threat and design patterns. Significant expert knowledge is required in the overall interdisciplinary security and domain specific challenges.

REFERENCES

[1] O'Halloran, B. M., Papakonstantinou, N., Giammarco, K., and Van Bossuyt, D. L., 2021. "A graph theory approach to predicting functional failure propagation during conceptual systems design". *Systems Engineering*.

[2] Papakonstantinou, N., Van Bossuyt, D. L., Linnosmaa, J., Hale, B., and O'Halloran, B., 2020. "Towards a zero trust hybrid security and safety risk analysis method". In ASME

2020 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers Digital Collection.

[3] Van Bossuyt, D. L., and Arlitt, R. M., 2020. "A functional failure analysis method of identifying and mitigating spurious system emissions from a system of interest in a system of systems". *Journal of Computing and Information Science in Engineering,* ***20***(5).

[4] Shane, S., and Sanger, D. E., 2011. "Drone crash in iran reveals secret us surveillance effort". *The New York Times,* ***7***.

[5] Ruegamer, A., Kowalewski, D., et al., 2015. "Jamming and spoofing of gnss signals–an underestimated risk?!". *Proc. Wisdom Ages Challenges Modern World,* ***3***, pp. 17–21.

[6] Zhu, H., Cummings, M. L., Elfar, M., Wang, Z., and Pajic, M., 2019. "Operator strategy model development in uav hacking detection". *IEEE Transactions on Human-Machine Systems,* ***49***(6), pp. 540–549.

[7] Warner, J. S., and Johnston, R. G., 2003. "Gps spoofing countermeasures". *Homeland Security Journal,* ***25***(2), pp. 19–27.

[8] Tippenhauer, N. O., Pöpper, C., Rasmussen, K. B., and Capkun, S., 2011. "On the requirements for successful gps spoofing attacks". In Proceedings of the 18th ACM conference on Computer and communications security, pp. 75–86.

[9] Kerns, A. J., Shepard, D. P., Bhatti, J. A., and Humphreys, T. E., 2014. "Unmanned aircraft capture and control via gps spoofing". *Journal of Field Robotics,* ***31***(4), pp. 617–636.

[10] Saulen, M. J., 2009. "The machine knows: what legal implications arise for gps device manufacturers when drivers following their gps device instructions cause an accident". *New Eng. L. Rev.,* ***44***, p. 159.

[11] Shen, J., Won, J. Y., Chen, Z., and Chen, Q. A., 2020. "Drift with devil: Security of multi-sensor fusion based localization in high-level autonomous driving under {GPS} spoofing". In 29th {USENIX} Security Symposium ({USENIX} Security 20), pp. 931–948.

[12] Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D., 2018. "Robust physical-world attacks on deep learning visual classification". In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1625–1634.

[13] Nassi, B., Mirsky, Y., Nassi, D., Ben-Netanel, R., Drokin, O., and Elovici, Y., 2020. "Phantom of the adas: Securing advanced driver-assistance systems from split-second phantom attacks". In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, pp. 293–308.

[14] Blanchard, B. S., Fabrycky, W. J., and Fabrycky, W. J., 1990. *Systems engineering and analysis*, Vol. 4. Prentice hall Englewood Cliffs, NJ.

[15] Stamatelatos, M., Dezfuli, H., Apostolakis, G., Everline, C., Guarro, S., Mathias, D., Mosleh, A., Paulos, T., Riha, D., and Smith, C., 2011. Probabilistic risk assessment procedures guide for nasa managers and practitioners. Tech. rep., NASA.

[16] Bowles, J. B., 1998. "The new sae fmeca standard". In Annual Reliability and Maintainability Symposium. 1998 Proceedings. International Symposium on Product Quality and Integrity, IEEE, pp. 48–53.

[17] Ericson, C. A., et al., 2015. *Hazard analysis techniques for system safety*. John Wiley & Sons.

[18] Van Bossuyt, D. L., O'Halloran, B. M., and Arlitt, R. M., 2019. "A method of identifying and analyzing irrational system behavior in a system of systems". *Systems Engineering,* ***22***(6), pp. 519–537.

[19] Aven, T., 2013. "On the meaning of a black swan in a risk context". *Safety science,* ***57***, pp. 44–51.

[20] National Institute of Standards and Technology (NIST), 2011. NIST Special Publication 800-39, Managing Information Security Risk. Tech. rep., March.

[21] International Electrotechnical Commission (IEC), 2009. IEC 62443 Industrial communication networks - Network and system security. Tech. rep., July.

[22] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), 2009. ISO/IEC 15408 Information technology — Security techniques — Evaluation criteria for IT security. Tech. rep., December.

[23] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), 2018. ISO/IEC 27000:2018 Information technology — Security techniques — Information security management systems. Tech. rep., February.

[24] Hacker, P., 2020. "A Legal Framework for AI Training Data". *Law, Innovation and Technology (forthcoming)*, March.

[25] Mohassel, P., and Zhang, Y., 2017. "Secureml: A system for scalable privacy-preserving machine learning". In 2017 IEEE Symposium on Security and Privacy (SP), IEEE, pp. 19–38.

[26] Al-Rubaie, M., and Chang, J. M., 2019. "Privacy-preserving machine learning: Threats and solutions". *IEEE Security & Privacy,* ***17***(2), pp. 49–58.

[27] Kreuk, F., Adi, Y., Cisse, M., and Keshet, J., 2018. "Fooling end-to-end speaker verification with adversarial examples". In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, pp. 1962–1966.

[28] Salem, A., Backes, M., and Zhang, Y., 2020. "Don't trigger me! a triggerless backdoor attack against deep neural networks". *ArXiv,* ***abs/2010.03282***.

[29] Sha, L., Camburu, O.-M., and Lukasiewicz, T., 2020.

Learning from the best: Rationalizing prediction by adversarial information calibration.
[30] Weng, C.-H., Lee, Y.-T., and Wu, S.-H. B., 2020. "On the trade-off between adversarial and backdoor robustness". *NeurIPS 2020*.
[31] Samek, W., Wiegand, T., and Müller, K.-R., 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models.
[32] Marino, D., Wickramasinghe, C., and Manic, M., 2018. "An adversarial approach for explainable ai in intrusion detection systems".
[33] Adadi, A., and Berrada, M., 2018. "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)". *IEEE Access,* ***6***, pp. 52138–52160.
[34] Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., and Herrera, F., 2020. "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai". *Information Fusion,* ***58***, pp. 82 – 115.
[35] Tsymbal, A., 2004. "The problem of concept drift: definitions and related work". *Computer Science Department, Trinity College Dublin,* ***106***(2), p. 58.
[36] Bhagoji, A. N., Chakraborty, S., Mittal, P., and Calo, S., 2019. "Analyzing Federated Learning through an Adversarial Lens". In Proceedings of the 36th International Conference on Machine Learning, Vol. 97 of *Proceedings of Machine Learning Research*, PMLR, pp. 634–643.
[37] Hooper, E., 2009. "Intelligent strategies for secure complex systems integration and design, effective risk management and privacy". *2009 3rd Annual IEEE Systems Conference*, pp. 257–261.
[38] Paté-Cornell, M., Kuypers, M., Smith, M., and Keller, P. N., 2018. "Cyber risk management for critical infrastructure: A risk analysis model and three case studies.". *Risk analysis : an official publication of the Society for Risk Analysis,* ***38 2***, pp. 226–241.
[39] Abdo, H., Kaouk, M., Flaus, J., and Masse, F., 2018. "A safety/security risk analysis approach of industrial control systems: A cyber bowtie - combining new version of attack tree with bowtie analysis". *Comput. Secur.,* ***72***, pp. 175–195.
[40] Haber, M., 2020. *Zero Trust*. 06, pp. 295–304.
[41] Samaniego, M., and Deters, R., 2018. "Zero-trust hierarchical management in iot". pp. 88–95.
[42] Tao, Y., Lei, Z., and Ruxiang, P., 2018. "Fine-Grained Big Data Security Method Based on Zero Trust Model". In 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS), pp. 1040–1045.
[43] Scott, B., 2018. "How a zero trust approach can help to secure your aws environment". *Network Security,* ***2018***, 03, pp. 5–8.
[44] Team, A.-I. Z. T. P., 2019. ZERO TRUST CYBERSECURITY CURRENT TRENDS, 04.
[45] Rose, S., Borchert, O., Mitchell, S., and Connelly, S., 2020. "Sp 800-207: Zero trust architecture". *National Institute of Standards and Technology (NIST), Computer Security Resource Center*, 08.
[46] Taffel, S., 2019. "Trusted machines? machine learning, more-than-human speed and dead labor in platform capitalism". *AoIR Selected Papers of Internet Research,* ***2019***, 10.
[47] Barzashka, I., 2013. "Are cyber-weapons effective? assessing stuxnet's impact on the iranian enrichment programme". *The RUSI Journal,* ***158***(2), pp. 48–56.
[48] Hirtz, J., Stone, R. B., McAdams, D. A., Szykman, S., and Wood, K. L., 2002. "A functional basis for engineering design: reconciling and evolving previous efforts". *Research in engineering Design,* ***13***(2), pp. 65–82.
[49] Cao, Y., Xiao, C., Cyr, B., Zhou, Y., Park, W., Rampazzi, S., Chen, Q. A., Fu, K., and Mao, Z. M., 2019. "Adversarial sensor attack on lidar-based perception in autonomous driving". In Proceedings of the 2019 ACM SIGSAC conference on computer and communications security, pp. 2267–2281.
[50] Davidson, D., Wu, H., Jellinek, R., Singh, V., and Ristenpart, T., 2016. "Controlling uavs with sensor input spoofing attacks". In 10th {USENIX} Workshop on Offensive Technologies ({WOOT} 16).
[51] Giray, S. M., 2013. "Anatomy of unmanned aerial vehicle hijacking with signal spoofing". In 2013 6th International Conference on Recent Advances in Space Technologies (RAST), IEEE, pp. 795–800.
[52] Donatti, M., Frazatto, F., Manera, L., Teramoto, T., and Neger, E., 2016. "Radio frequency spoofing system to take over law-breaking drones". In 2016 IEEE MTT-S Latin America Microwave Conference (LAMC), IEEE, pp. 1–3.

ACKNOWLEDGMENT

This research is partially supported by the VTT Technical Research Centre and the Naval Postgraduate School. Any opinions or findings of this work are the responsibility of the authors, and do not necessarily reflect the views of the sponsors or collaborators. The case study presented in this publication, while inspired by real systems, is intentionally fictional and idealized in nature. Approved for Public Release; distribution is unlimited.