A Survey of Security Threats to Intelligent Unmanned Systems

Chenhao Xu[1,2], Qiyang Han[1,2], Yingyu Chen[1,2], and Zhiwei Zhang[1,2]
[1]Shaanxi Key Laboratory of Network and System Security, Xidian University, Xi'an 710126, China
[2]School of Computer Science and Technology, Xidian University, Xi'an 710126, China

Intelligent unmanned systems are increasingly deployed in mission-critical scenarios, where attacks may enter through navigation signals, communication links, software interfaces, terminal devices, actuators, or human–machine interactions and propagate across sensing, computation and control, communication, actuation, and mission processes. Existing surveys are generally organized around specific platforms, components, or isolated security domains, making it difficult to characterize cross-layer propagation and the organizational relationships among multiple attack entries. This paper proposes a threat-entry-oriented multi-domain analysis framework for intelligent unmanned systems. Heterogeneous systems are mapped to a unified five-layer operational architecture, while attack impacts are projected into the physical, computing, network, and mission domains. Based on the number of actively manipulated entries and their entry-opening causal relationships, threats are classified into single-entry, parallel-entry, and cascading-entry threats. Using a structured narrative review, we synthesize academic research and representative real-world security incidents, distinguishing direct platform or platform-subsystem evidence from evidence transferred from related systems. Academic studies are organized according to single-entry, parallel-entry, and cascading-entry threat structures, while real-world incidents are analyzed in terms of attack entries, propagation mechanisms, cross-domain impacts, and system- or mission-level consequences. Based on the common characteristics of these threats, corresponding security requirements are derived for entry trustworthiness, cross-domain consistency, attack-chain containment, trust-boundary control, safe degradation, and trusted recovery.

***Index Terms*—Intelligent unmanned systems, threat-entry analysis, cross-layer propagation, cross-domain security, cyber-physical security, mission resilience.**

I. Introduction

UNMANNED systems are powered physical systems that operate without onboard human operators to accomplish designed missions [1]. With advances in perception, communication, computation, control, and autonomous decision-making, they are evolving from remotely operated or rule-driven platforms into intelligent unmanned systems with autonomous sensing, dynamic decision-making, closed-loop control, and cooperative execution [2], [3]. This paper focuses on mission-oriented systems including satellite and constellation systems, industrial unmanned systems, unmanned aerial vehicles (UAVs), ground and maritime platforms, and their swarms, and embodied robotic systems [4]–[7].

By integrating sensing, information processing, network communication, physical actuation, and multi-node coordination, intelligent unmanned systems can operate in complex and dynamic environments and support industrial production, military operations, space exploration, emergency response, logistics, and public services [3], [4]. Their growing role in continuous, real-time, and safety-critical missions has also made them strategically important to industrial development and national defense, as reflected in major policy initiatives in the United States, Europe, and China [8]–[11].

The security of intelligent unmanned systems is shaped by continuous interactions among the system, its physical environment, and human participants. Environmental signals affect sensing and state estimation, human commands and task interfaces influence mission objectives, and system actions in turn change the surrounding environment. Intelligence, interconnection, and autonomy therefore enlarge the attack surface and couple navigation signals, sensor inputs, communication links, software and model interfaces, remote-access channels, actuators, and human–machine interactions. Once local interference or manipulation is accepted by the system, its effects may propagate through internal dependencies and affect state estimation, control decisions, physical execution, collaboration, and mission completion [12], [13].

Recent incidents illustrate this propagation. Reports from the Ukraine conflict describe global navigation satellite system (GNSS) jamming that impairs unmanned-system navigation and mission execution [14]. The 2022 KA-SAT cyberattack disabled numerous satellite terminals and disrupted broadband services across Europe, as documented by both official disclosures and independent international reporting [15]–[17], while Starlink has faced electromagnetic interference and link suppression and commercial satellite user segments remain exposed to modem, management-interface, denial-of-service, and supply-chain risks [18], [19]. These cases show that attacks can begin at local physical, communication, or terminal entries and expand into cross-layer, cross-domain, and mission-level consequences. As heterogeneous platforms become more deeply interconnected, security threats can no longer be adequately understood from a single platform or isolated-domain perspective.

Existing surveys have examined UAV, satellite communication, industrial control, robotic and embodied AI, and cyber-physical system security [4], [6], [7], [13], [20]. Although they provide important platform- and technique-specific knowledge, most are organized around individual platforms, components, or isolated security domains. Consequently, they are less effective at explaining relationships among attack entries,

Manuscript received July 30, 2026; revised November 5, 2026. Corresponding author: Chenhao Xu (email: 2315348814@qq.com).

This work was supported in part by the Key R&D Program of Shandong Province, China (No. 2024CXPT039), in part by the Natural Science Basic Research Program of Shaanxi (Program No. 2026JC-YXQN-216).

the propagation of local impacts toward control and mission processes, or the cooperation and progressive expansion of multiple entries.

To address this limitation, this survey proposes a threat-entry-oriented multi-domain analysis framework. Heterogeneous intelligent unmanned systems are first abstracted into five operational layers: sensing, computation and control, communication, actuation, and coordination and mission. An attack entry is defined as an interface, channel, component, or interaction point through which an adversary actively influences the system, while its impacts are analyzed across the physical, computing, network, and mission domains. According to the number of active entries and their entry-opening causal relationships, threats are classified as single-entry, parallel-entry, and cascading-entry threats. The framework therefore explains where attacks enter, how multiple entries cooperate or progressively expand, and how local attacks evolve into system- and mission-level consequences.

The evidence reviewed in this paper consists of two complementary parts. Academic studies provide formal attack models, experimental mechanisms, and analytical methods, whereas real-world incidents reveal how attack entries, trust relationships, system dependencies, and recovery limitations interact in operational environments. The two forms of evidence are analyzed under the same threat-entry-oriented framework.

The main contributions of this paper are as follows:

- We propose a threat-entry-oriented analysis framework that maps heterogeneous intelligent unmanned systems to a unified five-layer operational architecture and jointly characterizes attack-entry locations, cross-layer propagation, multi-domain impacts, and mission-level consequences.
- We synthesize recent advances from academic research and representative real-world security incidents. Academic studies are reviewed according to single-entry, parallel-entry, and cascading-entry threat structures, while real-world incidents are comparatively analyzed through their initial entries, entry relationships, propagation paths, affected domains, and operational consequences.
- We summarize the common characteristics and protection priorities of the three threat structures and derive security requirements for entry trustworthiness, cross-domain consistency, attack-chain containment, trust-boundary control, safe degradation, and trusted recovery.

The remainder of this paper is organized as follows. Section I-A explains the survey methodology and evidence selection. Section II introduces the unified system abstraction and threat-entry-oriented multi-domain analysis framework. Sections III–V analyze single-entry, parallel-entry, and cascading-entry threats, respectively. Section VI summarizes security requirements, and Section VII concludes the paper and discusses future research directions.

A. Survey Methodology and Evidence Selection

This paper is a structured narrative survey with purposive, mechanism-oriented evidence selection. Its objective is to organize attack-entry relationships and derive security requirements, rather than estimate attack prevalence or provide an exhaustive census of the literature. The criteria and evidence labels below define the scope and evidential role of the selected material. The literature discovery process was not documented as a reproducible database search. Accordingly, this survey does not claim systematic-review completeness or report database hit counts or a screening flow diagram. Targeted checks of original publications and incident disclosures clarify the scope of selected claims.

The corpus has three roles: background surveys and definitions establish the system scope; attack and defense studies provide technical mechanisms; and incident reports provide operational observations. Standards and policy documents support context or security requirements and are not treated as attack demonstrations. For technical claims, the relevant evidence is the original research publication or the technical disclosure by an affected operator, incident-response team, or government security agency. News accounts provide incident context and corroboration, but do not by themselves establish an entry-opening relationship. Multiple reports of the same incident are treated as complementary accounts of one case rather than independent incidents.

A study is included in the threat synthesis when it identifies an actively manipulated signal, interface, component, or interaction; describes an attack mechanism or an explicit adversarial model; and supports analysis of an affected function, propagation path, or consequence. Defense studies qualify when their threat models identify the coordinated entries being defended against. An incident qualifies when a traceable public disclosure identifies the affected system and provides enough technical detail to analyze at least one entry or cross-domain effect. Purely generic security recommendations, unsupported attribution, and consequences without an identifiable mechanism are not used to establish a threat category. When a report supports an effect but not the causal relation among entries, it is retained only as qualified supporting evidence.

For each selected case, we record its source, target system, actively manipulated entry or entries, assumed attacker capabilities, entry-opening conditions, propagation mechanism, affected domains, and reported consequence. Tables III–VII summarize these fields and add an evidence-source column. D denotes direct evidence from an unmanned or robotic platform, or its platform-specific perception, localization, or control subsystem. D does not imply an end-to-end field trial or validation across all platform types. These labels describe evidence provenance rather than a ranking of study quality. T denotes evidence transferred from a related system: T-CPS for generic control or power systems, T-IT for general computing and software ecosystems, T-HMI for voice-assistant or human–machine interaction ecosystems, and T-INF for supporting communications or edge infrastructure. The target-system column identifies the actual evaluated or affected system.

Classification is applied to the analyzed attack scenario rather than permanently assigned to a publication. A single active entry remains single-entry when its effects propagate downstream without a new directly manipulated interface. Parallel-entry classification requires multiple entries whose ac-

tivation does not depend on capabilities obtained from another entry. Cascading-entry classification requires an identifiable transition that opens a further entry through newly acquired access, credentials, privileges, or trust. Timing, multiple affected domains, and a sequence of service calls are insufficient by themselves. Where the public record cannot resolve these conditions, the relationship is explicitly left unconfirmed. A publication may inform more than one scenario, but repeated discussion is not counted as independent evidence.

This selection favors informative mechanisms and publicly documented cases. It is subject to publication and disclosure bias, uneven platform coverage, and incomplete visibility into attacker access paths. The tables therefore support qualitative comparison, not frequency estimates or a ranking of attack likelihood. Section VI-D further specifies the limits of transferring related-system evidence to unmanned platforms.

II. Threat-Entry-Oriented Multi-Domain Analysis Framework for Intelligent Unmanned Systems

This section establishes a unified framework for analyzing how security threats enter, propagate through, and affect intelligent unmanned systems. As shown in Fig. 1, the framework is organized in a bottom-up manner. Shared system foundations support a unified five-layer operational structure. One or more actively manipulated entries are then identified from the operational structure and organized as single-entry, parallel-entry, or cascading-entry threats. Rather than applying the same propagation procedure to all three categories, the framework assigns a distinct analysis flow to each entry type. The resulting affected objects, propagation paths, interaction effects, and entry-opening relations are subsequently projected into four analytical domains. Finally, the affected-domain scope determines whether single-domain analysis or multi-domain joint analysis is required before system- and mission-level consequences are assessed.

The framework therefore separates four complementary analytical dimensions. The five-layer operational architecture identifies where system functions and actively manipulated objects are located. The attack-entry organization describes the relationships among one or more active entries. The entry-type-specific analysis flows determine how propagation evidence should be constructed for different entry relationships. The four analytical domains describe where the resulting security impacts are manifested and coupled. This separation avoids treating operational layers, analytical domains, attack entries, and threat categories as interchangeable classifications and provides a common basis for the analysis in Sections III–V.

A. Shared System Foundations and Unified Five-Layer Operational Architecture

Intelligent unmanned systems depend on a set of shared system foundations, including hardware, energy, computing resources, state data, and security mechanisms. These foundations jointly support sensing, computation, communication, physical execution, and mission coordination. They do not constitute an additional operational layer or an independent threat category. Instead, they provide the resources and operating conditions required by all upper-layer functions.

Although intelligent unmanned systems differ in platform form, deployment environment, and mission objective, their operational processes can generally be described through five functional layers: sensing, computation and control, communication, actuation, and coordination and mission. The sensing layer acquires environmental and system-state information. The computation and control layer performs state estimation, data fusion, decision-making, planning, and control generation. The communication layer supports command transmission, telemetry reporting, inter-node coordination, and cloud–edge–end information exchange. The actuation layer converts control outputs into physical actions and system-state changes. The coordination and mission layer organizes mission objectives, task allocation, human–machine interaction, service invocation, and multi-platform collaboration [2], [3], [13].

Satellite and constellation systems, industrial unmanned systems, UAV systems and swarms, and embodied robotic systems can all be mapped to this architecture, although their concrete components and operational processes differ [4]–[7], [20]. Satellite systems involve payload sensing, onboard computation, space–ground or inter-satellite communication, attitude and orbit actuation, and constellation-level missions. Industrial unmanned systems involve field sensing, PLC/RTU control, industrial communication, actuator operation, and process tasks. UAVs and embodied robots can likewise be described through onboard sensing, autonomous control, communication, physical actuation, and task coordination.

As summarized in Table I, each representative system occupies one row, while the five shared operational layers form the columns. The mapping does not imply that different systems use identical components or execution mechanisms. Instead, it identifies functionally corresponding positions in heterogeneous systems and preserves system-specific implementations within a common operational structure. For example, satellite payloads, industrial field sensors, UAV navigation devices, and robotic multimodal sensors differ substantially in physical form and operating conditions, but all perform sensing-layer functions. Similarly, constellation mission control, industrial production tasks, swarm missions, and human–robot collaboration all belong to the coordination and mission layer.

Potential attack-entry carriers are distributed across the five operational layers. Representative carriers include sensor or navigation inputs in the sensing layer; software, firmware, control logic, or intelligent models in the computation and control layer; links, protocols, routing functions, or remote-access interfaces in the communication layer; actuators or physical devices in the actuation layer; and commands, human–machine interfaces, task services, or collaborative interactions in the coordination and mission layer. However, a potential carrier is included in the actively manipulated entry set only when the adversary directly manipulates its data, state, privilege, interface, control behavior, or mission behavior.

The five-layer architecture therefore provides a common reference for locating active attack entries. It identifies where an actively manipulated object is positioned in the system's operational structure, but it does not restrict the subsequent

TABLE I
REPRESENTATIVE MAPPING OF INTELLIGENT UNMANNED SYSTEMS TO THE UNIFIED FIVE-LAYER OPERATIONAL ARCHITECTURE

System and Characteristic	Sensing	Computation and Control	Communication	Actuation	Coordination and Mission
Satellite systems Remote operation; long-delay and intermittent links	Optical and RF payloads	Onboard computer, GNC, and flight software	Space–ground and inter-satellite links	AOCS and propulsion	Constellation mission control
Industrial unmanned systems Process coupling; control tied to physical equipment	Field sensors and meters	PLCs, RTUs, and control logic	Fieldbus, TSN, and industrial 5G	Drives, valves, and robotic arms	Process and production tasks
UAV systems and swarms Mobile cooperation; dynamic topology and collaborative missions	GNSS, IMU, vision, and radar	Flight control and AI-based autonomy	C2, data, and inter-UAV links	Motors, servos, and flight surfaces	Flight and swarm missions
Embodied robotic systems Embodied interaction; multimodal sensing and human collaboration	Vision, audio, and tactile sensing	Middleware, intelligent models, and control policies	WLAN, edge, and cloud links	Locomotion and manipulation	Human–robot collaboration

Abbreviations: RF, radio frequency; GNC, guidance, navigation, and control; AOCS, attitude and orbit control system; PLC, programmable logic controller; RTU, remote terminal unit; TSN, time-sensitive networking; IMU, inertial measurement unit; C2, command and control; AI, artificial intelligence; WLAN, wireless local area network.

propagation scope. An attack entering through one layer may influence functions located in several other layers.

B. Attack-Entry Organization and Entry-Type-Specific Analysis Flows

A threat entry (also called an attack entry) is a functionally distinguishable interface, channel, component, or interaction point through which an adversary actively manipulates system data, states, privileges, control behavior, or mission behavior. The entry may be external, such as a navigation signal or wireless link, or internal, such as a software interface, model component, management function, or task-service interface.

The actively manipulated entry set contains all functionally distinguishable entries that are directly controlled or manipulated by the adversary during the analyzed attack. An affected component, layer, domain, or propagation channel is not regarded as an additional entry merely because it receives or transmits an attack effect. It becomes a new active entry only when the adversary actively manipulates its interface, privilege, state, control function, or service behavior. Obtaining access alone establishes a potential entry, which must be distinguished from an actively manipulated entry.

Based on the number of active entries and the entry-opening causal relationships among them, threats are organized as single-entry, parallel-entry, and cascading-entry threats. A single-entry threat contains only one actively manipulated entry. A parallel-entry threat contains two or more functionally distinguishable entries that independently support a shared attack objective. A cascading-entry threat begins from an initial entry and progressively opens one or more additional entries.

The decisive distinction between parallel-entry and cascading-entry threats is entry-opening causal dependence. Parallel entries do not require a foothold, privilege, credential, trust relationship, dependency position, or access capability obtained through another entry. They may be activated simultaneously, sequentially, or with partially overlapping effects. Temporal order alone therefore does not determine the category. By contrast, a later entry in a cascading-entry threat can be activated only because an earlier entry provides the required privilege, credential, trust relationship, dependency position, or access opportunity.

TABLE II
THREAT CLASSIFICATION AND ENTRY-TYPE-SPECIFIC ANALYSIS FLOWS

Threat Type	Entry Relationship and Identification Criterion	Entry-Type-Specific Analysis Flow
Single-entry	Only one functionally distinguishable entry is actively manipulated. Cross-layer or cross-domain propagation does not by itself create another entry.	Locate the active entry, trace single-source propagation, and identify affected objects.
Parallel-entry	Multiple entries support a shared objective, and none requires a capability obtained through another entry.	Separate the independent entries, trace their paths, analyze path convergence and interaction, and aggregate joint effects.
Cascading-entry	A later entry requires privileges, credentials, trust, dependencies, or access opportunities obtained through an earlier entry.	Analyze the initial entry, identify entry-opening conditions, activate the new entry, update the entry set, and iteratively trace propagation.

As shown in Fig. 1, the three threat categories share the same subsequent domain-projection and consequence-assessment framework, but they differ in how propagation evidence is constructed before domain projection.

a) Single-entry analysis: For a single-entry threat, the analysis first performs single-entry localization to determine the operational layer, component, interface, and function asso-

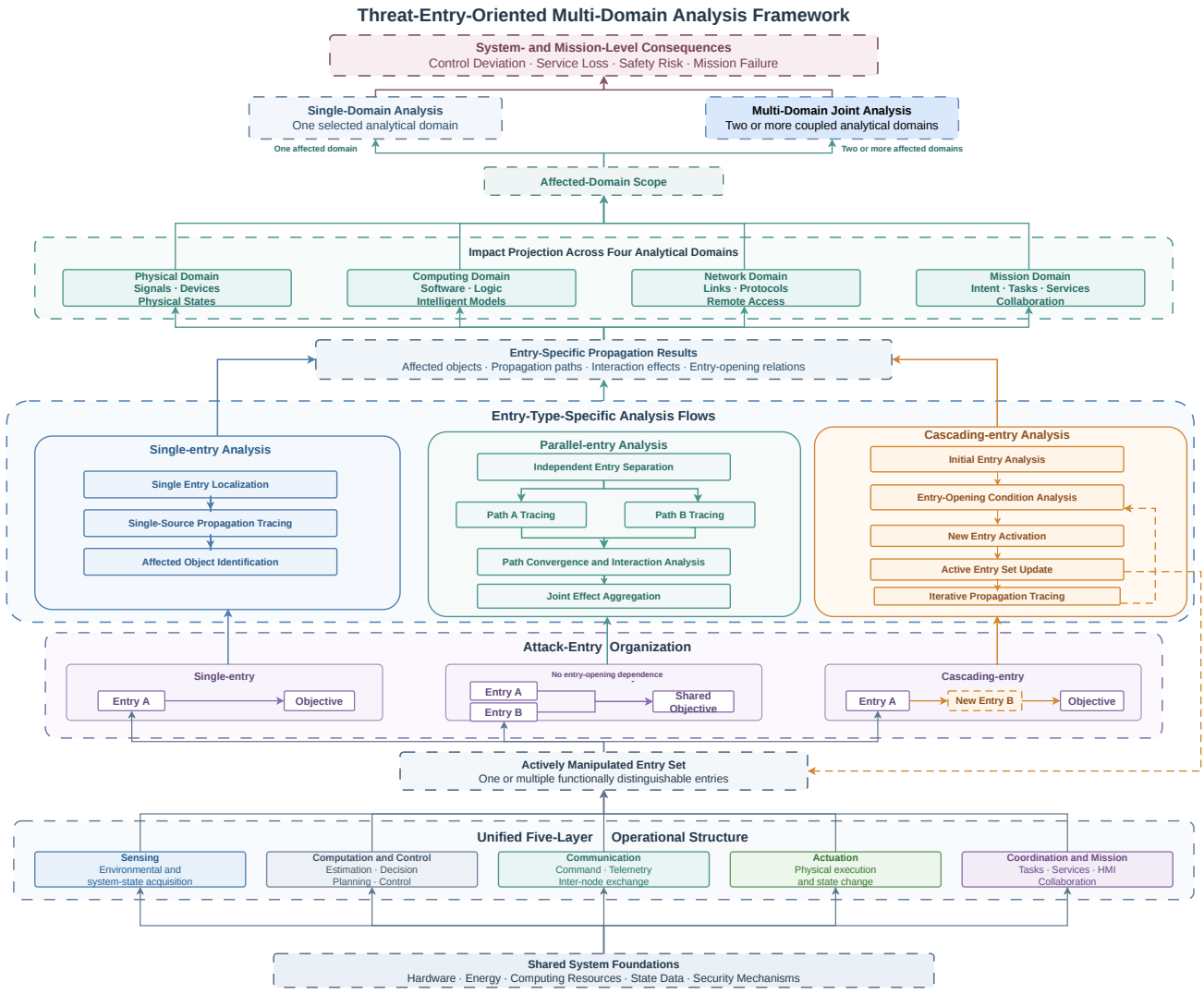


Fig. 1. Threat-entry-oriented multi-domain analysis framework with entry-type-specific analysis flows.

ciated with the only active entry. It then performs single-source propagation tracing, using the active entry as the unique starting point from which downstream effects are reconstructed. The tracing process follows the primary propagation path and any branches originating from the same source. Finally, affected-object identification determines which components, functions, states, services, or mission elements are influenced by the propagated effect.

For example, a navigation-spoofing attack may begin from a sensing-layer entry, corrupt state estimation in the computation and control layer, alter control outputs, and cause mission-state deviation. Although the effect crosses several layers and analytical domains, the threat remains single-entry as long as the adversary does not directly activate another functionally distinguishable entry.

b) Parallel-entry analysis: For a parallel-entry threat, the analysis first performs independent-entry separation. Each active entry is preserved as an independent propagation source rather than being prematurely merged into a single attack chain. The propagation path originating from each entry is then traced separately. After the individual paths have been reconstructed, path-convergence and interaction analysis determines whether they affect the same object, function, service, control process, or mission objective.

The interaction analysis considers whether independently generated effects reinforce, conceal, interfere with, or otherwise modify one another. Joint-effect aggregation then combines the individual and interactive effects into a unified representation of the parallel-entry threat. The defining property is that the entries may cooperate in achieving a shared objective while remaining causally independent with respect to entry activation.

c) Cascading-entry analysis: For a cascading-entry threat, the analysis begins with the initial active entry and identifies the capabilities produced by its compromise. Entry-opening condition analysis then determines whether these capabilities provide the privileges, credentials, trust, dependency position, or access opportunity required to activate another functionally distinguishable entry.

When an entry-opening condition is satisfied, the analysis

checks whether the adversary actively manipulates the newly accessible interface. Only then is it added to the actively manipulated entry set. Propagation tracing is then restarted or extended from the new entry. This update-and-trace process is performed iteratively because the newly activated entry may provide further capabilities that open additional entries. The iteration terminates when no further supported transition to an actively manipulated entry can be established within the analyzed scenario.

The feedback from a newly activated entry to the active-entry set is therefore specific to cascading-entry analysis. It represents expansion of the attack surface and attacker capability, rather than ordinary downstream propagation of an already manipulated state.

Although the three entry categories use different analysis flows, they rely on a common set of dependency relationships when reconstructing propagation. Data dependencies transmit manipulated observations, states, or messages to downstream functions. Control dependencies transmit abnormal decisions, commands, or feedback. Trust and privilege dependencies allow identities, permissions, credentials, or authority relationships obtained from one compromise to be reused elsewhere. Service and task dependencies propagate failures through required services, workflows, coordination relations, or mission processes.

The outputs of the three type-specific flows are normalized as entry-specific propagation results. These results contain four forms of information: affected objects, propagation paths, interaction effects, and entry-opening relations. Affected objects identify the components, functions, states, services, or mission elements influenced by the attack. Propagation paths describe how effects travel through system dependencies. Interaction effects describe convergence, reinforcement, concealment, interference, or other relationships among independently generated effects. Entry-opening relations record the causal conditions through which one active entry enables another. This unified result representation allows the three threat categories to enter a common multi-domain impact-analysis process without removing their category-specific characteristics.

C. Multi-Domain Impact Projection and Consequence Assessment

After the entry-specific propagation results have been constructed, the identified effects are projected into four analytical domains: the physical domain, computing domain, network domain, and mission domain.

The physical domain covers environmental signals, sensing objects, actuators, physical devices, and physical operating states. The computing domain covers software, firmware, middleware, control logic, hardware-assisted computation, and intelligent models. The network domain covers communication identities, protocol states, links, routing, remote access, and information exchange. The mission domain covers user intent, task semantics, service selection, authorization, workflow state, and collaborative mission execution.

The five-layer operational architecture and the four analytical domains serve different purposes. The five layers describe the functional organization of system operation and provide locations for identifying active entries and affected functions. The four domains describe where the resulting security impacts are manifested and coupled. They are complementary and non-exclusive rather than strictly one-to-one.

For example, a sensing-layer component may simultaneously involve physical signals, embedded computation, and network communication. Likewise, an attack entering through a software interface in the computation and control layer may produce physical-domain and mission-domain consequences. Therefore, the four domains are not used as a static threat taxonomy. They provide analytical spaces for describing how security effects appear and interact after propagating through the operational structure.

This distinction also separates attack-entry organization from affected-domain scope. The number of active entries does not determine the number of affected domains. A single-entry threat may propagate into several analytical domains, whereas a parallel-entry or cascading-entry threat may remain concentrated within one domain. Entry classification is determined by actively manipulated entries and their entry-opening relationships, while domain scope is determined by where the resulting impacts are manifested.

After domain projection, the framework determines the affected-domain scope. When the identified effects are confined to one analytical domain, single-domain analysis is performed by examining the affected objects, propagation paths, and security properties within that selected domain. When the identified effects involve two or more coupled analytical domains, multi-domain joint analysis is required. The joint analysis examines how effects transfer, interact, reinforce, or constrain one another across domain boundaries.

For example, navigation spoofing may first produce a physical-domain signal abnormality, then lead to a computing-domain estimation error, and finally cause mission-domain trajectory deviation. Such a threat requires multi-domain joint analysis even when it originates from only one active entry. By contrast, multiple independent software interfaces may be manipulated in parallel while their effects remain confined to the computing domain. Such a threat is multi-entry but may still be analyzed primarily within one domain.

The final stage summarizes system- and mission-level consequences. Control deviation refers to system behavior departing from the intended control result. Service loss refers to the degradation or unavailability of required functions. Safety risk represents potential harm to equipment, operators, users, or the physical environment. Mission failure indicates that the system can no longer satisfy its intended task or collaborative objective.

Accordingly, the complete framework follows a bottom-up analytical logic: shared system foundations support the five-layer operational structure; active entries are identified and organized according to their entry-opening relationships; each entry type is processed through a distinct analysis flow; the resulting propagation evidence is projected into the four analytical domains; affected-domain scope determines whether single-domain or multi-domain joint analysis is required; and the final effects are summarized as system- and mission-level

consequences.

Based on this framework, Sections III–V analyze single-entry, parallel-entry, and cascading-entry threats, respectively. In particular, Section III focuses on single-source propagation from one active entry, Section IV analyzes convergence and interaction among independent propagation paths, and Section V examines entry-opening conditions and iterative attack-surface expansion. Section VI then derives the corresponding security requirements.

III. Single-Entry Threats

A single-entry threat occurs when an attacker actively exploits only one distinguishable attack entry to affect an intelligent unmanned system. The attack entry may involve an external physical signal, communication protocol, software or hardware interface, intelligent model component, or task interaction. The term "single-entry" describes the number of actively manipulated attack entries, rather than the scope of the resulting impact. The sensing, computation and control, communication, actuation, and coordination and mission layers are interdependent. An anomaly introduced through one attack entry may therefore propagate across the unified five-layer architecture and ultimately affect system control and mission execution. Table III summarizes representative single-entry threats across the four domains. The following subsections analyze their mechanisms and propagation characteristics using the unified architecture.

A. Single-Entry Threats in the Physical Domain

Physical-domain entries are mainly located at the boundary between the sensing layer and the external environment. They include navigation signals, laser echoes, acoustic signals, optical signals, and other physical quantities detectable by sensors. Such attacks typically do not directly modify system code or communication protocols. Instead, they alter how the system obtains environmental states, causing the sensing layer to produce erroneous inputs while operating normally.

Intelligent unmanned systems typically assume a stable correspondence between sensor outputs and the real environment. Navigation receivers convert received signals into position and time information [21]. LiDAR converts optical echoes into spatial point clouds [22], [23], while microphones convert acoustic or optical disturbances into audio inputs for speech systems [24], [25]. Downstream modules focus on how these data are fused and used, but often lack continuous verification of signal provenance and physical origin. Once an attacker constructs an input that satisfies a sensor's response conditions, the system may derive an erroneous sensing estimate without any device failure.

GNSS spoofing illustrates how this trust relationship extends into state estimation. Multisensor fusion is commonly used to improve localization robustness. However, fusion algorithms assess plausibility mainly from numerical relationships among observations and cannot readily determine whether a signal was generated by an attacker. Shen et al. showed that a slowly varying spoofing signal can gradually bias the fused localization estimate [21]. The resulting position error resembles a continuous state change rather than an obvious anomaly. The computation and control layer then interprets the biased estimate as the true position and uses it for path planning and motion control. A single physical-domain attack entry can thus propagate from corrupted sensing to deviation of the mission trajectory.

LiDAR attacks demonstrate the conversion risk between physical signals and digital representations of the environment. Cao et al. generated false obstacles in point clouds using malicious laser echoes [22]. Sun et al. further exploited inadequate modeling of occlusion and point-cloud structure to conduct black-box attacks against object detectors [23]. The attacker manipulates the optical input received by the sensor. After entering the computation and control layer, however, malicious echoes become spatial points that satisfy the required data format. The object detector may misinterpret injected points as real objects when their point-cloud features satisfy its learned decision criteria. The risk therefore extends beyond the physical signal itself because the injected input is processed as if it were legitimate sensor data.

Acoustic and optical injection attacks further show that a sensor's actual response boundary may exceed the input range assumed during system design. DolphinAttack exploits nonlinear effects in microphone circuits to recover inaudible ultrasonic signals as valid voice commands [24]. Light Commands uses modulated lasers to induce equivalent audio inputs in microphones [25]. CommanderSong hides malicious instructions in ordinary audio, creating a discrepancy between machine recognition and human auditory judgment [26]. Although these attacks manipulate different physical carriers, all exploit a mismatch between sensor responses and the user's intended utterance. After recognition, the coordination and mission layer may assign task semantics to the anomalous input and trigger device actions or service calls.

Within the unified five-layer architecture, the distinctive security assumption of the physical domain is environmental authenticity. A physical-domain attack entry manipulates an external signal so that the sensing layer produces an erroneous digital representation. Downstream modules may continue to follow their intended logic because the injected input appears sensor-valid. Protection must therefore verify whether sensing data remain consistent with the physical environment, rather than relying only on sensor availability or data-format validity.

B. Single-Entry Threats in the Network Domain

Network-domain attack entries lie in the communication layer and at its interfaces with the computation and control layer and the coordination and mission layer. They mainly include wireless signals, protocol messages, user-plane traffic, and remote communication interfaces. In intelligent unmanned systems, communication links not only transmit data but also maintain state consistency between controllers and actuation platforms, individual platforms and collaborating nodes, and edge nodes and cloud services. Once a communication attack entry is manipulated, the system loses more than link availability. It may also lose the ability to verify communication identities, message content, and interaction states.

TABLE III
REPRESENTATIVE STUDIES OF SINGLE-ENTRY THREAT SCENARIOS

Domain	Method	Target System	Attack Entry	Mechanism	Main Impact	Evidence Source
Physical	Gradual GNSS Spoofing [21]	Autonomous-driving localization	GNSS signal	Gradual position bias	Localization drift; control deviation	D
Physical	LiDAR Sensor Attack [22]	Driving perception	Laser echo	False-point injection	False obstacles; abnormal braking	D
Physical	Black-Box LiDAR Spoofing [23]	Driving object detection	Optical input	Occlusion exploitation	Detection failure	D
Physical	DolphinAttack [24]	Voice-control system	Ultrasonic signal	Inaudible command injection	Unauthorized execution	T-HMI
Physical	Light Commands [25]	Voice-control system	Laser signal	Laser-based audio injection	Unauthorized commands	T-HMI
Physical	CommanderSong [26]	Speech recognition	Audio input	Adversarial command embedding	Hidden-command recognition	T-HMI
Network	LTEInspector [27]	4G LTE network	Protocol message	State-machine exploitation	Identity confusion; denial of service	T-INF
Network	IMP4GT [28]	LTE user plane	User-plane traffic	Integrity-bypass impersonation	Traffic tampering; impersonation	T-INF
Network	Signal Overshadowing [29]	LTE network	Wireless signal	Synchronized signal overpowering	Broadcast-message tampering	T-INF
Computing	Robot Controller Attack [30]	Industrial robot	Software interface	Control-logic modification	Motion deviation; safety violation	D
Computing	Malicious ROS Node [31]	Raven-II robotic system	ROS node	State-triggered behavior	Control disruption; task failure	D
Computing	Thunderclap [32]	Operating system/IOMMU	DMA peripheral	IOMMU isolation bypass	Memory access; kernel compromise	T-IT
Computing	CLKSCREW [33]	Processor/trusted execution	DVFS interface	Voltage/frequency fault injection	Trusted-execution breach	T-IT
Computing	Trojaning Attack [34]	Neural-network model	Model/training interface	Trojan implantation stage	Attacker-specified prediction	T-IT
Computing	Blind Backdoors [35]	Machine-learning pipeline	Training code/loss	Backdoor implantation stage	Triggered malicious behavior	T-IT
Mission	Skill Squatting [36]	Voice-skill ecosystem	Skill invocation	Invocation confusion	Service redirection	T-HMI
Mission	Command Hijacking [37]	Voice-controlled IoT	Skill invocation	Malicious-skill interception	Unauthorized device control	T-HMI
Mission	Dangerous Skills [38]	Voice-assistant ecosystem	Skill invocation	Squatting and masquerading	Impersonation; data disclosure	T-HMI

LiDAR: light detection and ranging; LTE: Long-Term Evolution; ROS: Robot Operating System; DMA: direct memory access; IOMMU: input-output memory management unit; DVFS: dynamic voltage and frequency scaling. D: direct platform or platform-subsystem evidence. T: transferred evidence; HMI = voice interaction, INF = supporting infrastructure, IT = general computing. Impacts refer to the reported target system, not all unmanned platforms. Classification concerns the listed scenario and assumes the stated initial access; a wider acquisition chain may be cascading. Backdoor rows cover implantation; active trigger delivery through another interface requires separate entry analysis.

Communication protocols typically reduce complexity through layered design, with each protocol layer validating only the information available locally. The physical layer determines whether a signal can be demodulated, the data-link layer handles frames and connections, and the network layer manages addressing and forwarding. Upper-layer services then interpret application content within established sessions. This structure creates interlayer trust: once a lower-layer message is accepted, upper-layer modules rarely revalidate its physical origin or the complete interaction process. Attacks targeting LTE protocol states, user-plane integrity, and low-level wireless-message reception demonstrate how this trust can be exploited [27]–[29].

LTEInspector's analysis of LTE authentication, attachment, paging, and state transitions showed that state-machine flaws in protocol specifications can be triggered by specific message sequences. An attacker need not control the entire cellular network. By changing a terminal's judgment of its current communication state through one protocol interaction entry, the attacker may cause identity confusion, location exposure, or denial of service. For unmanned platforms that rely on cellular communication for remote control, telemetry, and task updates, abnormal protocol states may further disrupt control sessions and task synchronization, turning a communication-layer problem into control and mission risks [27].

IMP4GT revealed a disconnect between control-plane authentication and user-plane trustworthiness. Even after mutual authentication between a terminal and the network, missing user-plane integrity protection may allow traffic manipulation and bidirectional impersonation. The attack enters through a single communication link but affects both the data-link and network layers of the protocol stack. This changes how upper

layers assess traffic provenance. When control commands, state data, and coordination information depend on this trust, impersonation is no longer merely a form of network deception. It may directly alter the control objectives and state awareness received by the platform [28].

The physical signal overshadowing attack proposed by Yang et al. shows that a network-domain entry may lie at the bottom of the protocol stack. By transmitting a precisely synchronized malicious wireless signal with dominant power, an attacker causes a terminal to receive a tampered broadcast message. Although the attack carrier is a physical wireless signal, its primary targets are the communication protocol and connection state, so it belongs to the network domain in the four-domain framework. This attack shows that the legitimacy of communication content cannot be determined solely by whether a wireless signal can be demodulated normally. Once a low-level wireless input alters a protocol message, the upper-layer state machine may still process the incorrect content through its normal workflow [29].

The distinctive security assumptions of the network domain concern identity, protocol state, and message integrity. A manipulated communication-layer trust decision may distort data, commands, or identity information before they reach computation and mission functions. UAV swarms, satellite constellations, and distributed robots are especially exposed because they depend on inter-node information sharing. Network-domain protection must therefore go beyond connectivity and encryption. It must verify the mutual consistency of communication identities, protocol states, message content, platform states, and task relationships.

A complementary direction is to derive device-trust evidence from the temporal characteristics of network access. ATCAuth extracts timing patterns from device–access-point interactions and uses them as software-observable fingerprints for preliminary device identification before full WLAN access is granted [39]. Although the method was evaluated in general WLAN environments rather than on unmanned platforms, it demonstrates how access behavior can complement static identifiers and credentials when assessing whether a communicating endpoint is consistent with its claimed identity.

C. Single-Entry Threats in the Computing Domain

Computing-domain entries are mainly located in the computation and control layer, including controller software, operating systems, robotic middleware, peripheral interfaces, hardware management mechanisms, and components used to train and deploy intelligent models. Physical- and network-domain attacks often influence decisions by corrupting information received by the system, whereas computing-domain attacks act directly on the internal logic by which data are interpreted, fused, and converted into control actions.

Numerous software and hardware components jointly form the trusted computing base of an intelligent unmanned system. Sensor drivers receive state data, middleware organizes inter-module communication, controllers generate actuation commands, operating systems manage resources and permissions, and intelligent models perform recognition, planning, and decision-making. These components do not operate in isolation but span sensing, control, and mission execution. Representative attacks against these components include controller manipulation, malicious middleware nodes, peripheral and hardware-management attacks, and model backdoors [30]–[35]. Once a computing-domain attack entry is exploited, malicious logic may be incorporated into normal data and control flows and executed with legitimate system privileges.

Research on industrial robot controllers demonstrates the direct link between software attack entries and physical actuation. Quarta et al. found that controller vulnerabilities can modify motion parameters, control logic, and safety configurations. Although the attack occurs in the computation and control layer, it ultimately causes motion deviations, reduced work quality, or violations of safety constraints. The controller converts computation results into actuation commands within the unified five-layer architecture. Once its software state is manipulated, the actuation layer may still treat malicious instructions as legitimate control outputs [30].

Robotic middleware extends this risk to component collaboration. Chung et al. deployed a malicious ROS node in a Raven-II surgical robot. The node used control data shared through the middleware to identify task phases and trigger malicious behavior at critical moments. Although the attack entry was only one software node, it could observe system state, select trigger conditions, and influence the control process through normal messaging mechanisms. Middleware is designed to reduce module coupling and support functional expansion, but it also allows malicious components with legitimate communication capabilities to blend more easily into the system's operational chain [31].

Peripheral interfaces and hardware management mechanisms show that the computing-domain boundary extends beyond the traditional software boundary. Thunderclap uses malicious DMA peripherals to access shared memory and exploits IOMMU isolation flaws to read sensitive data, modify kernel state, and hijack control flow [32]. On unmanned platforms integrating cameras, radar, communication modules, and edge-computing devices, peripherals often have broad data-access privileges, so their trustworthiness directly affects the operating system and control software. CLKSCREW induces processor faults through software-controlled voltage and frequency interfaces. The attacker enters through energy-management functions, but the resulting computation errors can breach privilege and trusted-execution boundaries. Software–hardware management interfaces thus become important computing-domain entries connecting resource management with security control [33].

The introduction of intelligent models extends computing-domain entries from program execution logic to training and decision logic. Liu et al. implanted Trojans through model modification or training interfaces, allowing a model to behave normally on ordinary inputs but produce attacker-specified outputs under particular trigger conditions [34]. Bagdasaryan and Shmatikov showed that an attacker need only control the training code or loss function to implant a covert backdoor without significantly changing performance on the normal task. Once deployed, the malicious logic becomes part of the

computation process and is repeatedly invoked by recognition, planning, and task modules [35].

For these backdoor studies, the single-entry scenario isolates model or training-code manipulation during implantation. If an attacker later supplies a trigger through a distinct input interface, the full scenario contains an additional active entry. Its classification then depends on whether activating that interface requires capabilities acquired through implantation; temporal order or trigger dependence alone is insufficient.

These attacks reveal differences between computing-domain threats and conventional software vulnerabilities. Some entries do not cause an immediate system crash, but instead alter the mapping from system states to behavior. A malicious robot node waits for a critical task phase, a backdoored model behaves abnormally only when a trigger appears, and a hardware fault attack may target a security-sensitive computation. Such anomalies are strongly state-dependent. They may appear to be intermittent errors, model misclassifications, or device faults, while the underlying cause is active manipulation of internal processing logic.

The difficulty of exposing state-dependent failures becomes more pronounced in swarm systems, where a failure may emerge only from a particular combination of environmental inputs, agent states, and cooperative behaviors. RSFuzz addresses this problem through a robustness-guided swarm-fuzzing framework based on behavioral constraints, targeting the large compositional input space and dynamically evolving cooperative states of multi-robot swarms [40]. This work is primarily a vulnerability-discovery and robustness-testing approach rather than an online attack-detection mechanism, but it provides direct evidence that swarm-level behavioral constraints can guide the discovery of latent logical failures in unmanned-system software.

The distinctive security assumption of the computing domain is the integrity of processing logic. Computing-domain threats may alter software, hardware, middleware, peripheral access, or model behavior while preserving syntactically valid data and commands. Protection must therefore extend beyond code and firmware integrity to the complete processing chain. This chain includes middleware components, software–hardware management interfaces, and the training, deployment, and operation of intelligent models.

D. Single-Entry Threats in the Mission Domain

Mission-domain attack entries are mainly located in the coordination and mission layer, including task requests, service invocation, task-object selection, permission granting, and third-party service interfaces. Unlike threats in the physical, network, and computing domains, mission-domain attacks do not necessarily compromise sensors, communication protocols, or internal processing logic. Instead, they manipulate the mapping from user intent to the actual executor, causing legitimate task requests to be received, interpreted, or executed by an incorrect service.

Representative studies mainly focus on third-party applications and services in voice-assistant platforms [36]–[38]. In such systems, users express task intent through voice commands, after which the platform performs speech recognition, service matching, permission checking, and device invocation. Attackers can exploit weaknesses in service names, invocation phrases, interaction identities, or permission mechanisms to redirect legitimate requests to malicious services. The user's intended objective may remain unchanged, while the actual executor is replaced by an attacker-controlled entity.

Skill Squatting exploits similarities in skill names or pronunciations to replace the legitimate service intended by the user with a malicious skill [36]. The attack entry lies not in the acoustic signal itself, but in the platform's skill-naming and service-matching mechanisms. Once invoked, the malicious skill may provide false results, manipulate subsequent interactions, or obtain information originally intended for the legitimate service.

Command Hijacking further demonstrates that malicious third-party skills can intercept legitimate commands intended for smart devices, system functions, or connected services [37]. By manipulating skill invocation and command handling, an attacker redirects a valid task request to an attacker-controlled service and then abuses existing permissions to affect device operation. A normal task invocation may therefore propagate through service selection, permission use, and device control, ultimately resulting in unauthorized execution.

Dangerous Skills reveals additional risks arising from voice squatting and voice masquerading [38]. A malicious skill may hijack the invocation of a legitimate service at the beginning of a task or impersonate that service during subsequent interaction, causing both the user and the platform to misidentify the actual executor. Such attacks undermine consistency among task intent, service identity, and execution permissions, potentially resulting in service impersonation, sensitive-information disclosure, or unauthorized device operation.

The triggering of task execution does not necessarily mean that the initial attack entry belongs to the mission domain. If an attacker manipulates ultrasonic, laser-based, or other acoustic or optical signals, the entry remains in the physical domain. A threat belongs to the mission domain only when the attacker directly manipulates task semantics, service selection, skill invocation, or permission relationships. The core security assumption of the mission domain is therefore consistency among user intent, the selected executor, and granted permissions. Protection should focus on task-provenance verification, service-identity authentication, invocation control, and least-privilege enforcement.

IV. Parallel-Entry Threats

A parallel-entry threat occurs when an attacker actively manipulates two or more functionally distinguishable entries toward a shared objective, without one entry supplying the access capability needed to activate another. The entries may be activated simultaneously or sequentially; overlap in time is not a classification requirement. Their effects must jointly contribute to the analyzed objective. If an earlier entry supplies a required foothold, credential, privilege, or trust relationship for a later entry, that relationship is cascading. Functional distinguishability identifies separate manipulated interfaces;

TABLE IV
REPRESENTATIVE STUDIES OF CROSS-DOMAIN PARALLEL-ENTRY THREATS

Domain Combination	Method	Target System	Attack Entries	Mechanism	Main Impact	Evidence Source
Physical–Network	Cross-Layer LQG Attack [41]	Networked control system	Physical disturbance; channel interference	State disruption; observation blocking	Estimation error; control degradation	T-CPS
Physical–Network	Physical–DoS Attack [42]	Cyber-physical power system	Equipment attack; denial of service	Physical damage; feedback loss	Persistent deviation	T-CPS
Physical–Computing	Bilevel Coordinated Attack [43]	Power system	Line attack; data tampering	Fault concealment; false dispatch	Operational risk; cascading failure	T-CPS
Physical–Computing	CCPA [44]	Smart grid	Physical attack; PMU injection	Signature cancellation	Detection failure; erroneous control	T-CPS
Physical–Computing	MTD-Based Localization [45]	Power grid	Line disconnection; false data	Reactance perturbation; localization	Attack exposure; outage localization	T-CPS
Physical–Computing	Game-Theoretic MTD [46]	Power system	Line disconnection; measurement tampering	Reactance perturbation	Improved detection; defense cost tradeoff	T-CPS
Physical–Computing	Secured-PMU Defense [47]	Power system	Line attack; measurement tampering	PMU placement; constrained dispatch	Outage prevention	T-CPS

LQG: linear-quadratic-Gaussian; DoS: denial of service; CCPA: coordinated cyber-physical attack; MTD: moving-target defense; PMU: phasor measurement unit. All rows are T-CPS: transferred evidence from control or power systems. Defense-oriented rows contribute explicit threat models, not additional operational incidents. Corresponding access and closed-loop dependencies must be validated on an unmanned platform.

entry-opening causal independence determines whether they are parallel. Neither cross-domain propagation nor simultaneous effects alone establish additional independent entries.

Table IV summarizes representative studies of cross-domain parallel-entry threats. The representative literature reviewed in this survey contains relatively few direct studies of coordinated parallel-entry attacks on intelligent unmanned platforms. This section therefore also includes evidence from closely related cyber-physical systems that share closed-loop state estimation, communication, control, and physical actuation. Existing work focuses mainly on physical–network and physical–computing combinations. Defensive studies are included when their threat models explicitly characterize coordinated attacks involving multiple actively manipulated entries. Publicly reported incidents are analyzed separately in the network–computing subsection.

The selected cyber-physical evidence is mapped by functional analogy, not experimental equivalence. Power-grid line states motivate comparison with actuator, propulsion, or energy-component states; PMU measurements motivate comparison with position, attitude, or equipment-health observations; and dispatch decisions motivate comparison with planning or control. All studies in Table IV are transferred evidence (T-CPS). Their grid topology, observability assumptions, attacker access, and control time scales do not establish feasibility or performance on mobile unmanned platforms. The unmanned-system scenarios below are implications of the proposed mapping, not results reported by those studies.

A. Physical–Network Cross-Domain Entries

Physical–network parallel-entry threats jointly manipulate the real operating state and the communication-based observation or feedback process. The physical attack directly changes the system state, whereas the network attack reduces the availability, timeliness, or trustworthiness of the information needed to detect and correct that change.

Huang et al. developed a game-theoretic model of cross-layer coordinated attacks on cyber-physical systems. An attacker can disturb the physical process while simultaneously interfering with the observation channel. The physical attack directly changes the system state, whereas communication interference affects observation availability and state-estimation accuracy. The two strategies are interdependent: the physical-layer attack-defense state influences the timing of communication interference, while estimation errors caused by interference degrade control performance. This work reveals the interdependence between physical-state manipulation and communication-observation blocking from a dynamic-game perspective [41].

Tian et al. further studied coordinated cyber-physical attacks involving denial of service. An attacker physically disrupts power-system lines or equipment through a physical-domain attack entry while using a network-domain attack entry to block selected measurements and communication data. In covert scenarios, denial of service reduces the valid measurements available to defenders and makes the physical attack harder to identify accurately. In non-covert scenarios, denial of service and physical damage jointly increase operating deviations. Simulations showed that coordinated cross-domain attacks can cause more serious effects than either physical attacks or denial of service alone while consuming fewer attack resources [42].

The risk of such threats arises not merely from the simultaneous occurrence of two attacks, but from the network entry changing the conditions under which the control system responds to a physical anomaly. After a physical disturbance causes the real state to deviate, missing, delayed, or discarded observations increase uncertainty in state estimation. The

controller cannot promptly distinguish normal fluctuations, device faults, and malicious disturbances, nor can it generate effective compensation commands from an accurate state. As feedback control degrades, the initial physical deviation may accumulate during continued operation and further affect actuation stability and mission completion [41], [42].

In UAVs and embodied robots, this mechanism may involve external disturbance of platform motion or actuators together with suppression of telemetry, sensor networks, or remote-control links. In satellites and industrial unmanned equipment, device states may be altered while state reporting and remote intervention are weakened. Here, the network attack is not a consequence of the physical attack but a second entry independently manipulated by the attacker. It reduces the visibility of the physical anomaly, prolongs the anomalous state, and weakens control and recovery mechanisms.

Physical–network cross-domain parallel-entry threats disrupt the correspondence among real states, sensing information, and control feedback. The sensing layer may acquire incomplete information, and the communication layer may fail to deliver it promptly. The computation and control layer must then infer system state from limited observations. Even when each module follows its prescribed logic, the closed loop may diverge from the intended mission because the real state remains partially unobserved.

B. Physical–Computing Cross-Domain Entries

Physical–computing cross-domain parallel-entry threats manipulate both the real physical state and its digital representation. A physical-domain attack entry changes equipment, lines, energy components, or actuation processes. The domain assigned to false-measurement injection depends on where the attacker performs the manipulation. Manipulation during sensor-signal formation belongs to the physical domain, while modification of packets in transit belongs to the network domain. Tampering with a database, state-estimation input, or digital model belongs to the computing domain. Here, the computing-domain designation emphasizes the corrupted digital representation used for estimation and control in the reviewed models. The actual injection interface must be identified separately; its delivery channel may also involve the physical or network domain.

In the unified five-layer architecture, the computation and control layer does not directly observe the real environment. It reconstructs platform state from sensing data, communication information, and system models. Consistency between the physical state and its digital representation is fundamental to state estimation, fault diagnosis, and control decisions. If an attacker changes a physical object while synchronously modifying its digital representation, a persistent discrepancy can be created between the real system and computational awareness. Sensors may continue to produce data and state estimators may converge normally, yet the computational result no longer corresponds to the actual operating state.

Li et al. modeled this coordinated attack as a bilevel optimization problem. The attacker changes the actual power-system topology through physical line attacks while conducting topology-preserving and load-redistribution data attacks. The topology and measurements received by the control center therefore appear normal. The computing-domain attack entry both conceals physical line faults and may induce erroneous dispatch. This combination can amplify local physical anomalies and increase cascading-failure risk [43].

Deng et al. examined how false measurements can cancel the observable signature of a physical attack. Their method constructs a false-data injection vector from phasor measurement unit data. The physical attack entry alters the real system state, while the computing-domain attack entry reshapes the digital state observed by the control center. When these entries are coordinated, conventional bad-data detection may find no clear inconsistency. The control system may then continue estimation, control computation, and operational scheduling based on an incorrect system-state estimate [44].

From a defensive perspective, Chen et al. examined how changes in physical–digital coupling can expose and localize coordinated attacks. Their threat model combines physical line disconnection with synchronized measurement tampering. Distributed flexible AC transmission devices actively change line reactance, invalidating false data constructed from the attacker's original parameters. A deep-learning method then localizes the disconnected line. Although the study focuses on defense, its threat model shows that concealment requires the computing-domain attack entry to track changes in the physical state [45].

Lakshminarayana et al. likewise used moving-target defense to alter the coupling between the physical system and measured data. By perturbing transmission-line reactance, the defender invalidates the attacker's prior system knowledge. The effectiveness of a cross-domain parallel-entry threat therefore depends on more than independent access to two attack entries. It also requires an accurate model of the coupling between the physical and computing domains. When the perturbation invalidates the attack model under the evaluated conditions, measurement tampering may no longer conceal the physical anomaly [46].

Huang et al. extended this line of work to the system-level consequences of unsafe dispatch and cascading outages. Their secure phasor measurement unit placement strategy does not seek to eliminate every attack. Instead, it prevents coordinated line attacks and measurement tampering from inducing line overloads and outages through an incorrect system view. The study shows that cross-domain parallel-entry threats may not only conceal existing faults but also amplify physical damage through otherwise normal control processes [47].

Coordination between physical and computing attack entries creates an attacker-controlled discrepancy between the physical world and its digital representation. The physical entry changes actual operating conditions, while the computing entry preserves apparent normality or induces an erroneous control decision. Commands generated from the corrupted state estimate may satisfy program logic and data-format requirements but remain unsuitable for the real environment. The anomaly can persist across state estimation, control decisions, and actuation feedback. It may consequently expand a local device fault into operational instability, erroneous dispatch, or mission failure [43]–[47].

In intelligent unmanned systems, such a threat may involve damaging actuators, propulsion components, or energy systems while falsifying equipment-health indicators, attitude estimates, and fault-diagnosis information. The platform may then fail to identify the damage and continue its mission or issue control actions that increase risk. The key risk of a physical–computing cross-domain threat is therefore not merely physical damage or data tampering, but the attacker's active control over how the system perceives reality through two separate entries.

C. Network–Computing Cross-Domain Entries

The incidents examined here provide qualified combined-effect evidence concerning communication services, control functions, and terminal or host operational states. Their public disclosures do not establish whether the relevant attack capabilities were acquired through independent entries. Network-domain actions may disrupt connectivity or manipulate control communications, while computing-domain actions may erase data, corrupt configurations, or disable hosts. These combined effects can impair system operation and recovery, but confirmed parallel-entry classification requires evidence of entry independence.

Intelligent unmanned systems generally depend on remote management, network configuration, and software-defined terminals for large-scale operation. A network anomaly alone may cause only a temporary interruption, after which a terminal can reconnect. If the terminal's internal state is damaged, however, it may remain inoperable after communication services recover because critical data, configurations, or storage states have been corrupted. When network and computing entries act together, a short communication interruption can become persistent system incapacitation, and recovery expands from link reconnection to terminal reset, system reinstallation, or hardware replacement.

Table V summarizes KA-SAT and Industroyer2 as qualified combined-effect evidence from supporting infrastructure and a power system, respectively.

The KA-SAT cyberattack on 24 February 2022 exhibited behavior consistent with coordinated network- and computing-domain effects in a satellite communication ground-and-terminal system. Independent reporting by the Associated Press documented the disabling of tens of thousands of modems across Europe and the loss of remote connectivity to wind turbines, while Reuters later reported the coordinated attribution of the operation to Russia by the United States, the United Kingdom, Canada, Estonia, and the European Union [16], [17]. Viasat reported malicious traffic that hindered legitimate terminals from remaining online and an intrusion through a misconfigured VPN appliance into a trusted management network, where destructive commands overwrote critical flash-memory data on numerous user modems [15]. SentinelLabs subsequently reported that AcidRain, a MIPS-targeting wiper, was used to damage the storage and operational state of affected embedded terminals [48].

Traffic-based denial of service and terminal-state destruction affected different functions, but the public account does not establish that their capabilities were acquired through independent entries. We therefore use KA-SAT as supporting-infrastructure evidence (T-INF) of combined effects, not as a confirmed parallel-entry attack on an unmanned platform. The reported VPN-to-management-to-modem progression separately supports an entry-opening chain within the destructive branch. Viasat reported no direct compromise of the satellite itself. This distinction prevents an observed satellite-service outage from being interpreted as demonstrated onboard control compromise [15], [48].

The Industroyer2 operation provides transferred power-system evidence (T-CPS) of coordinated control actions and host destruction. ESET reported IEC-104 control functionality together with CaddyWiper and Linux/Solaris wiping components, and interpreted the destructive activity as an attempt to obstruct recovery [49]. Contemporary reporting described the attempted power disruption as thwarted [50], [51]. These observations support the combined failure mechanism, but do not establish independent acquisition of the control and wiping entries. The case is therefore not counted as confirmed parallel-entry evidence, and it does not directly validate an attack on an unmanned vehicle or robot.

These combined effects motivate protection of connectivity, terminal state, and recovery capability. Network attacks obstruct control, state reporting, and remote management. Computing-domain damage can prevent affected devices from reconnecting even after communication services recover. For satellite terminals, UAV ground-control equipment, industrial edge nodes, and robot control hosts, this combination may cause widespread disconnection, configuration damage, and prolonged interruption of mission services. It can also substantially increase the time and resources required for recovery.

TABLE V
REAL-WORLD INCIDENTS SUPPORTING COMBINED NETWORK AND COMPUTING EFFECTS

Domain Combination	Incident and Date	Target System	Reported Actions	Main Impact	Evidence Source
Network–Computing	KA-SAT, 24 Feb. 2022 [15]–[17], [48]	Consumer satellite broadband terminals	Traffic-based denial of service; VPN intrusion, management access and flash-data destruction	Broadband outage; terminal reset or replacement	T-INF; parallel relation unconfirmed
Network–Computing	Industroyer2, 8 Apr. 2022 [49]–[51]	Power provider and substations	IEC-104 control actions; Windows and Linux/Solaris wiping components	Attempted power disruption; recovery obstruction	T-CPS; parallel relation unconfirmed

These incidents illustrate combined effects, not confirmed independent parallel entries. KA-SAT concerns supporting infrastructure; Industroyer2 concerns a power system. Neither is direct evidence of an end-to-end unmanned-platform attack.

V. Cascading-Entry Threats

A cascading-entry threat occurs when an adversary first exploits an initial attack entry and subsequently opens one or more additional active entries through acquired privileges, credentials, trust relationships, software dependencies, infrastructure dependencies, or task interactions. The defining characteristic is that a later entry depends on a capability obtained through an earlier entry.

Cascading-entry threats differ from both single-entry and parallel-entry threats. A single-entry threat contains only one actively manipulated entry, although its effects may propagate across several operational layers or analytical domains. A parallel-entry threat contains multiple entries that independently support a shared objective. In a cascading-entry threat, however, a later entry becomes available only because an earlier compromise provides the required privilege, credential, trusted position, dependency relationship, or access opportunity.

According to the analysis flow introduced in Section II, cascading-entry analysis begins with the initial active entry. The capabilities obtained through this entry are then examined to determine whether they satisfy the conditions for activating another functionally distinguishable entry. Once a new entry is activated, it is added to the actively manipulated entry set, and propagation tracing continues from the new entry. This process is repeated until no additional entry-opening relationship can be identified.

Table VI distinguishes direct cascading-entry evidence from qualified task-chain analogies. It records the actual target system, interfaces, supported mechanism, and evidence source. Table VII separately summarizes transferred incident evidence, including access capabilities for which a complete downstream platform attack chain has not been established.

A. Privilege- and Trust-Driven Cascading Threats

Privilege- and trust-driven cascading threats occur when an adversary uses an initial foothold to obtain higher privileges or abuse a trusted operational relationship. The initial entry is not the final attack objective. Instead, it changes which engineering tools, management interfaces, credentials, control functions, or safety mechanisms become reachable.

A later interface may still accept the adversary's operation as legitimate because the adversary uses credentials, identities, or authority obtained through the initial compromise. Examining only the final malicious action may therefore fail to reveal how the attack became possible. Analysis must reconstruct the transition from the initial foothold to the newly reachable privileged interface.

The TRITON/TRISIS incident provides representative real-world evidence of this mechanism. Public analysis showed that the attack framework was designed to interact with Triconex safety instrumented system controllers [54]. From a cascading-entry perspective, the critical progression was from an engineering or remote-access environment to the safety-controller programming interface.

The engineering or remote-access environment constituted the initial entry. The trusted engineering relationship then provided the authority required to activate a controller-level entry. After this transition, the attack boundary expanded from an ordinary host or management environment to the safety protection layer, where controller logic and protection behavior could be manipulated.

This progression is relevant to intelligent unmanned systems because such systems rely on remote-maintenance interfaces, ground-control stations, fleet-management consoles, engineering tools, cloud-management services, and safety controllers. An adversary may compromise a maintenance interface and then reuse the acquired authority to reach flight-control parameters, robotic-controller configurations, satellite-terminal management functions, propulsion settings, or actuator-control logic.

The main security problem lies in the entry-opening transition. Detection therefore requires evidence of credential use, privilege changes, trusted-channel access, remote-session establishment, and subsequent operations on control interfaces.

TABLE VI
Direct cascading-entry evidence and qualified task-chain analogies

Domain Combination	Study	Target System	Entries or Interfaces	Supported Mechanism	Reported Impact	Evidence Source
Network–Computing	SROS 2 Supply-Chain Attack [52]	QCar2 / SROS 2 testbed	Third-party package; stolen keystore; control topics	Credential theft enables authenticated re-entry	Spoofed control or perception	D
Mission–Computing	Skill Squatting [36]	Voice-assistant ecosystem	Skill invocation; conditional later service access	Redirection demonstrated; additional entry requires evidence	Service redirection	T-HMI; conditional chain
Mission–Computing	Dangerous Skills [38]	Voice-assistant ecosystem	Skill invocation; interaction trust	Squatting and masquerading; new permission not assumed	Impersonation; data disclosure	T-HMI; conditional chain
Mission–Computing	SkillExplorer [53]	Third-party skill ecosystem	Skill dialogue	Behavioral privacy analysis; no entry-opening chain established	Private-information requests; continued listening	T-HMI; mechanism reference

SROS 2 denotes Secure Robot Operating System 2. D denotes the platform-specific SROS 2 proof of concept. T-HMI rows motivate task-chain analysis, but do not demonstrate cascading attacks on unmanned platforms. Invocation, continued interaction, or data disclosure alone does not establish a new active entry.

B. Supply-Chain and Update-Channel Cascading Threats

Supply-chain and update-channel cascading threats exploit dependencies among software producers, third-party components, build environments, update channels, and downstream systems. Intelligent unmanned systems increasingly depend on firmware updates, over-the-air updates, AI-model updates, robotic middleware packages, third-party libraries, task plug-ins, and cloud-delivered configurations.

The SROS 2 study summarized in Table VI demonstrates this mechanism in a robotic environment [52]. A compromised third-party package provides the initial entry. After execution, the adversary obtains keystore credentials and uses them to enter authenticated ROS 2 control topics.

The control-topic injection is a new active entry because the adversary uses stolen credentials to publish spoofed messages through a distinct trusted interface. The actively manipulated entry set therefore expands from the software package to the control or perception topic. Credential possession alone would establish access capability rather than an additional active entry.

The SolarWinds SUNBURST incident illustrates a similar progression at a larger scale. The adversary compromised the SolarWinds Orion build or update process and distributed a trojanized update through a trusted software channel [55]. The supplier-side build environment served as the initial entry, while downstream customer networks and management environments became newly opened entries.

The 3CX incident demonstrates a multi-stage supply-chain relationship. Public analysis linked the compromise of 3CX to malicious software obtained from another vendor [56]. The upstream software compromise opened access to the 3CX development environment. This environment then opened the trusted 3CX build and distribution channel, which exposed downstream endpoints.

NotPetya shows how an update entry can further open credential- and network-based entries. Its initial delivery was linked to the compromised M.E.Doc update mechanism [57], [58]. After execution, credential reuse and vulnerability exploitation enabled propagation to additional internal hosts and services.

These cases share a common progression. An upstream dependency is compromised, a trusted build or update channel delivers malicious code, the code executes in a downstream environment, and additional credentials, services, or control interfaces become reachable.

For intelligent unmanned systems, this progression may affect onboard computers, edge nodes, ground stations, fleet-management platforms, cloud-based mission services, robotic middleware, and model-deployment environments. Software provenance must therefore be verified across build, signing, distribution, installation, credential use, and runtime interaction.

C. Cloud–Edge–End Dependency Cascading Threats

Cloud–edge–end dependency cascading threats arise from the distributed architecture of modern intelligent unmanned systems. Such systems depend on cloud services, edge gateways, routers, firewalls, ground stations, remote-maintenance terminals, telemetry aggregators, and fleet-management platforms.

A compromised supporting node may provide stored credentials, network visibility, management relationships, or routing capabilities that enable access to additional systems. The compromised node therefore becomes more than an affected component. It becomes a position from which new entries can be activated.

VPNFilter demonstrates this risk in edge infrastructure. It affected routers and network-attached devices and supported traffic monitoring, credential theft, command-and-control communication, proxying, and destructive behavior [59]. Compromise of an edge device could therefore open additional entries related to credentials, traffic observation, and downstream network access.

Cyclops Blink was described as a modular malware framework deployed on network devices, with persistent access, file download and execution, and the ability to add modules [60]. These capabilities support the concern that a compromised infrastructure node can sustain and extend attacker access. The cited advisory does not, however, establish a complete downstream attack chain into an unmanned platform. Such a chain is a transfer hypothesis requiring evidence of a further reachable interface and the capability used to manipulate it.

Volt Typhoon-related advisories document persistent access to critical-infrastructure IT networks and preparation for possible disruption of operational technology [61]. Reported activities include credential abuse, lateral movement, and living-off-the-land techniques. These observations support analysis of access expansion and pre-positioning; they do not establish completed operational disruption or an attack chain into an unmanned platform.

In intelligent unmanned systems, a similar chain may begin from a remote-access interface or edge device, continue through a gateway or management node, and then reach an operational network, edge-computing platform, telemetry service, fleet-management function, or device-control interface.

A compromised ground station may expose multiple unmanned platforms. A compromised gateway may expose telemetry and local control services. A compromised cloud-management account may enable modification of task plans, software images, models, or permissions across multiple devices.

The central risk is therefore not only infrastructure unavailability. The more important risk is that the attacker can reuse the trusted position, credentials, and dependencies of the compromised node to activate additional entries.

Table VII summarizes the real-world incidents discussed in the preceding subsections.

D. Task-Chain and Ecosystem Cascading Mechanisms and Qualified Analogies

A task-chain or ecosystem cascade can arise when an initial task interaction enables an attacker to manipulate a distinct additional interface. The initial interaction may involve a task

TABLE VII
TRANSFERRED INCIDENT EVIDENCE FOR CASCADING MECHANISMS AND ACCESS CAPABILITIES

Domain Combination	Incident and Date	Target System	Reported Progression	Impact or Limitation	Evidence Source
Network–Computing	TRITON/TRISIS, 2017 [54]	Safety instrumented system	Engineering foothold; controller programming	Safety-control manipulation; physical risk	T-CPS
Network–Computing	SolarWinds SUNBURST, 2020 [55]	Enterprise software supply chain	Trojanized update; downstream access	Persistent organizational access	T-IT
Network–Computing	NotPetya/M.E.Doc, 2017 [57], [58]	Enterprise and industrial networks	Malicious update; credential reuse; lateral movement	Destructive disruption	T-IT
Computing	3CX, 2023 [56]	Software distribution ecosystem	Upstream software compromise; build and distribution access	Downstream endpoint exposure	T-IT
Network–Computing	VPNFilter / Cyclops Blink, 2018 / 2022 [59], [60]	Edge-network infrastructure	Device compromise; modular capabilities; persistence	Infrastructure access; further platform chain not established	T-INF
Network–Computing	Volt Typhoon, 2024 [61]	Critical-infrastructure IT/OT	IT foothold; credential abuse; lateral access	Persistence; prospective disruption risk	T-INF

All rows are transferred evidence. Dates identify the incident or public disclosure discussed here. Infrastructure capabilities do not establish a complete attack path to an unmanned platform; further entry-opening transitions require case-specific evidence.

request, service invocation, third-party skill, plugin, or human–machine interaction. A cascading classification requires evidence that this interaction opens further access through newly acquired credentials, permissions, or service capabilities.

This mechanism is relevant to unmanned systems that use open task ecosystems, reusable mission modules, cloud-based planning services, or natural-language interfaces. Publicly documented operational incidents establishing such cascades remain limited, so the following discussion draws on the voice-assistant studies in Table VI. Skill Squatting and Dangerous Skills are treated as conditional task-chain analogies, whereas SkillExplorer serves as a mechanism reference. These T-HMI studies support analysis of task-interaction risks without establishing cascading attacks on unmanned platforms.

Skill Squatting demonstrates that invocation errors can redirect a request to an attacker-controlled skill [36]. This directly supports the invocation-manipulation scenario discussed in Section III-D. A cascading interpretation requires additional evidence that the initial interaction obtains a new credential, permission, or service capability and that the attacker then actively uses a distinct interface. Service redirection alone does not demonstrate this transition. Its use here is a transferred task-chain analogy (T-HMI), not an independently validated cascading attack on an unmanned platform.

Dangerous Skills shows how voice squatting and masquerading can redirect interactions and mislead users about the responding service [38]. This supports a task-trust failure in a voice-assistant ecosystem. Whether subsequent data collection constitutes another active entry depends on the actual permission and interface transition. If it merely continues through the same malicious skill under existing authority, the scenario remains single-entry. Extension to robotic task execution therefore requires evidence of newly opened control or service access.

SkillExplorer analyzes third-party skill behavior and documents requests for private information and continued listening after stop commands [53]. It provides transferred evidence of interaction and privacy risks (T-HMI); it does not by itself demonstrate privilege escalation, a newly opened device-control entry, or an end-to-end robotic attack chain. Its contribution to this subsection is to motivate scrutiny of task-service behavior. A cascading interpretation remains conditional on a separately established entry-opening transition.

This mechanism can be mapped to intelligent unmanned systems. A robot may load a third-party task skill, an unmanned vehicle may invoke a cloud-based planning service, and a UAV swarm may install a mission-specific plugin. The initial invocation may subsequently open access to mission data, external services, control messages, or physical execution. Such extensions require platform-specific evidence that the initial interaction enables manipulation of a distinct interface.

The core security property is consistency among mission intent, selected service identity, granted permissions, external dependencies, and actual execution behavior. Authentication of the initial task request is insufficient when the invoked service can later acquire new permissions or activate additional capabilities without revalidation.

Across the four progression mechanisms, cascading-entry threats share the same fundamental structure. The initial entry is not the final attack boundary. Instead, the adversary uses the initial compromise to change which credentials, privileges, services, devices, interfaces, or dependencies become reachable.

Privilege- and trust-driven progression moves from an ordinary foothold to a higher-privilege control interface. Supply-chain progression moves from upstream dependencies to trusted distribution channels and downstream platforms. Cloud–edge–end progression moves through gateways, infrastructure nodes, operational networks, and end devices. Task-chain progression qualifies as cascading only when an earlier compromise enables active manipulation of a distinct later interface. Service invocation, permission use, data access, or physical execution alone does not establish that transition.

These mechanisms are particularly relevant to intelligent unmanned systems because such systems are heterogeneous, interconnected, update-dependent, service-oriented, and mission-critical. Protection therefore requires not only detection of the initial compromise, but also continuous iden-

tification of entry-opening conditions, changes in the actively manipulated entry set, and expansion of adversary capabilities. The corresponding requirements for attack-chain containment, trust-boundary control, safe degradation, and trusted recovery are derived in Section VI.

VI. Security Requirements Analysis

The security requirements in this section are derived from the failure mechanisms identified in the preceding threat analysis. A single-entry threat shows how a locally anomalous signal, message, component, or task interaction can acquire downstream trust. A parallel-entry threat shows how several locally plausible entries can conceal or reinforce one another when their effects are evaluated separately. A cascading-entry threat shows how an initial foothold changes credentials, privileges, or dependencies and thereby opens later entries. Finally, mission-compromising effects show why detection alone is insufficient when normal operation can no longer continue safely.

This derivation connects each threat structure to four questions: what problem is exposed, what evidence is required, what the protection mechanism must achieve, and how the system should respond. Table VIII makes this reasoning explicit. The resulting requirements—entry trustworthiness, cross-domain consistency, attack-chain containment, trust-boundary control, safe degradation, and trusted recovery—are therefore tied to the reviewed attack mechanisms rather than appended as general recommendations. The Govern, Identify, Protect, Detect, Respond, and Recover functions of the NIST Cybersecurity Framework 2.0 provide a lifecycle reference for implementing them [62].

A. Entry Trustworthiness and Cross-Domain Consistency

Single-entry cases reveal a transfer-of-trust problem. Spoofed sensor inputs, protocol-valid messages, compromised processing logic, and malicious service invocations can all satisfy local checks before being consumed by downstream control or mission functions. Entry trustworthiness therefore requires evidence of signal or message provenance together with consistency between the reported state and the platform's physical, computational, and mission context. If provenance cannot be established or the states conflict, the protection goal is not merely to label an anomaly, but to prevent it from retaining full authority. The system should reduce confidence, restrict high-risk actions, and adopt conservative control until the entry is revalidated. Continuous verification and least-privilege access support this treatment of uncertain inputs [63].

At the device level, multidimensional intrinsic identity construction and dynamic seamless authentication provide a related direction for maintaining trust after initial access [64]. Such approaches are relevant to mobile and intermittently connected unmanned nodes because a one-time credential check cannot establish that the same trustworthy device and operating context persist throughout a mission. However, evidence from general IoT environments should be treated as an architectural reference rather than direct validation on unmanned platforms.

For human-operated control terminals, entry trust may also depend on whether the current interaction behavior remains consistent with that of the authorized operator. SPUAuth demonstrates that fine-grained behavioral biometrics derived from motion-sensor and gesture characteristics can complement knowledge-based credentials and improve resistance to observed-secret imitation [65]. Although the method targets smartphone pattern unlocking rather than unmanned-system control stations, it provides transferable evidence for combining explicit credentials with continuous behavioral verification at human–machine interaction entries.

Parallel-entry mechanisms and qualified combined-effect evidence motivate cross-domain consistency checks. In a parallel-entry scenario, one manipulated entry may suppress the evidence needed to detect another. Physical damage paired with falsified measurements can preserve a normal-looking digital state. Communication disruption combined with terminal destruction can also complicate diagnosis and recovery, even when the relationship between their entries remains unconfirmed. Detection must therefore correlate trusted timestamps, state transitions, control outputs, actuation feedback, and mission events across domains. The decisive evidence is temporal and causal inconsistency at the system level, not necessarily an extreme value at one component. Once locally plausible evidence becomes globally irreconcilable, suspected entries should be isolated and their combined influence restricted in control decisions. Asset identification, segmentation, controlled remote access, and continuous monitoring provide the operational basis for this response [5].

Causal analysis provides one possible mechanism for converting such cross-layer observations into root-cause evidence. In a complex 5G environment, Xu *et al.* constructed a Bayesian causal model to uncover causal links across multilayer network and service data and thereby identify the root causes of

TABLE VIII
Security requirements derived from threat structures

Threat Structure	Exposed Problem	Required Evidence	Detection/Protection Goal	Response Requirement
Single-entry	Local anomaly gains downstream trust	Signal provenance, state consistency	Reject or reduce trust in anomalous entry	Conservative control
Parallel-entry	Multiple entries conceal each other	Cross-domain temporal and causal evidence	Detect globally inconsistent but locally valid states	Isolate coordinated entries
Cascading-entry	Initial foothold opens later entries	Credential, privilege and dependency transitions	Identify entry-opening behavior	Revoke trust and interrupt chain
Mission-compromising effects	Critical operation cannot continue normally	Mission state and minimum functional set	Preserve controllability	Safe degradation and trusted recovery

anomalies [66]. Although the study addresses 5G network operations rather than unmanned-system attacks directly, its method-level evidence is relevant to unmanned platforms that depend on cellular and edge services: correlated alarms should be interpreted through cross-layer causal dependencies instead of being treated as independent component anomalies.

B. Attack-Chain Containment and Trust-Boundary Control

Cascading-entry threats are identified by entry-opening behavior: the initial compromise changes which accounts, devices, services, or control functions become reachable. The required evidence is therefore a transition history of credentials, privileges, software provenance, trust relationships, and service dependencies. A later action extends the entry chain when it actively manipulates a distinct interface using authority or access created by an earlier compromise.

Semantic provenance graphs offer a concrete representation for preserving and analyzing this transition history. Han *et al.* construct semantic provenance graphs from multisource logs, extract graph-structural representations, and combine static features with dynamic behavioral evidence for early ransomware detection [67]. Their evaluation concerns ransomware on conventional computing hosts rather than intelligent unmanned systems; nevertheless, the provenance-graph approach illustrates how processes, files, communications, and operations can be linked into causal behavior chains. For unmanned systems, the same principle would require extending the graph with sensor observations, control commands, actuator responses, node identities, and mission-state transitions.

Containment should act at the transition that expands attacker capability. Management, control, and mission-execution privileges should be separated, and high-impact operations should be revalidated against device state and mission phase. Abnormal credential use, privilege escalation, or changes in update provenance should trigger identity revocation, node isolation, or suspension of critical control access. Software, firmware, models, and task plugins likewise require verification across build, signing, distribution, and deployment; otherwise, a trusted update path becomes the mechanism that opens entries on multiple platforms. These controls interrupt the chain by withdrawing newly acquired trust rather than assuming that the initial boundary will always hold [68], [69].

C. Mission-Oriented Safe Degradation and Trusted Recovery

Mission-compromising effects require the response to be evaluated against the mission state, not only the health of individual components. Immediate shutdown may be unsafe for a vehicle in motion, a robot performing a critical operation, or a distributed system that still retains partial control. The system must therefore identify its minimum functional set—the trusted sensing, communication, computation, and actuation capabilities needed to remain observable and controllable—and preserve that set while disabling uncertain functions.

Safe degradation converts this evidence into bounded operation. Declining sensing confidence may require conservative control and reduced autonomy; communication anomalies may require a verified local mode; suspicious computation or mission components may require suspension of high-privilege actions. The specific action is platform-dependent, but its goal is consistent: prevent irreversible effects without discarding the control needed for recovery.

Trusted recovery requires evidence that the identified attack path has been removed or contained. Reconnecting a device or restarting software is insufficient if compromised credentials, models, configurations, permissions, or task states are restored unchanged. Recovery should begin from a verified baseline, rebuild identities and permissions, confirm physical and mission-state consistency, and only then resume higher-risk functions. In this way, safe degradation preserves controllability during the incident, while trusted recovery reduces the risk of re-establishing the original single-entry, parallel-entry, or cascading-entry attack structure [70].

D. Evidence Transferability and Validation Limits

The evidence-source labels constrain the scope of the preceding requirements. Direct evidence is concentrated in autonomous-driving subsystems, robot controllers and middleware, and the SROS 2 testbed; these studies do not collectively validate the framework across UAV swarms, maritime platforms, or satellite constellations. Conversely, the parallel-entry studies in Table IV are drawn from generic control and power systems, while most incident evidence for cascading mechanisms concerns enterprise or infrastructure systems. Their inclusion establishes useful failure mechanisms in the reported environments, not their prevalence or demonstrated exploitability on unmanned platforms.

A transfer is defensible only when the target platform has the relevant functional dependency and the attacker can obtain the required access. Grid-based concealment models require corresponding sensing and control dependencies, but grid observability, dispatch intervals, and line-reactance controls cannot be assumed for a moving robot. Enterprise supply-chain and credential-abuse cases require comparable update trust, stored secrets, or remote-management paths; they do not establish that flight-critical partitions or safety controllers are reachable. Voice-assistant cases require comparable task matching, service identities, and authorization semantics; a misleading interaction does not by itself imply physical actuation. Infrastructure outages establish service-dependency risk but do not prove onboard compromise.

Platform-specific validation should therefore state the attacker capabilities, accessible entries, causal transition evidence, autonomy level, mission phase, timing constraints, and safety mechanisms. Simulation and testbed results should be distinguished from operational observations. Outcomes should include control deviation, detection delay, mission completion, safe-degradation behavior, and recovery cost under the tested conditions. Until such validation is available, the transferred cases support design hypotheses and protection priorities; their attack success rates and defense performance must not be carried over as quantitative guarantees.

VII. Conclusion

This survey proposed a threat-entry-oriented framework for security analysis of intelligent unmanned systems. The framework maps heterogeneous platforms to five operational layers and distinguishes these functional locations from four impact domains. Threats are classified by the number of active entries and their entry-opening causal relationships: single-entry threats involve one actively manipulated entry; parallel-entry threats involve multiple entries without entry-opening dependence; and cascading-entry threats involve a later entry enabled by an earlier compromise. Temporal order or cross-domain propagation alone does not determine the category.

The structured narrative synthesis distinguishes direct platform or platform-subsystem evidence from mechanisms transferred from related systems. It illustrates how attacks can propagate, conceal one another, or expand access, without establishing their prevalence across unmanned platforms. Protection priorities include entry trustworthiness, cross-domain consistency, attack-chain containment, trust-boundary control, safe degradation, and trusted recovery. Transferred evidence supports these qualitative requirements but does not establish platform-specific exploitability or quantitative defense performance.

Future work should validate the framework on diverse platforms using explicit attacker capabilities, causal transition evidence, and mission-level outcomes. Common benchmarks should compare propagation paths, control deviation, detection delay, mission completion, and recovery cost. Detection should relate sensing, communication, software execution, and task semantics rather than treat their anomalies independently. Large models, tool invocation, mission plugins, and cloud–edge–end dependencies also require lifecycle evaluation. AI risk-management and secure-system-engineering standards provide relevant foundations [71]–[76], while experimental validation remains necessary before transferred mechanisms can support operational assurance.

References

[1] H.-M. Huang, Ed., "Autonomy levels for unmanned systems (ALFUS) framework, volume I: Terminology, version 2.0," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 1011-I-2.0, 2008. [Online]. Available: https://www.nist.gov/document/nistsp1011-i-2-0pdf

[2] E. R. Griffor, C. Greer, D. A. Wollman, and M. J. Burns, "Framework for cyber-physical systems: Volume 1, overview," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 1500-201, 2017. [Online]. Available: https://doi.org/10.6028/NIST.SP.1500-201

[3] H. Wang, H. Zhao, J. Zhang, D. Ma, J. Li, and J. Wei, "Survey on unmanned aerial vehicle networks: A cyber physical system perspective," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1027–1070, 2020, doi: 10.1109/COMST.2019.2962207.

[4] M. Kang, S. Park, and Y. Lee, "A survey on satellite communication system security," *Sensors*, vol. 24, no. 9, Art. no. 2897, 2024, doi: 10.3390/s24092897.

[5] K. Stouffer, M. Pease, C. Tang, T. Zimmerman, V. Pillitteri, S. Lightman, A. Hahn, S. Saravia, A. Sherule, and M. Thompson, "Guide to operational technology (OT) security," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 800-82 Revision 3, Sep. 2023, doi: 10.6028/NIST.SP.800-82r3.

[6] F. Abdullayeva and O. Valikhanli, "A survey on UAVs security issues: Attack modeling, security aspects, countermeasures, open issues," *Control Cybern.*, vol. 52, no. 4, pp. 405–439, 2023, doi: 10.2478/candc-2023-0044.

[7] J. Duan *et al.*, "Embodied intelligence security with vision-language models: A survey," *Mach. Intell. Res.*, vol. 23, pp. 746–769, 2026, doi: 10.1007/s11633-025-1626-x.

[8] J. Clark, "Defense innovation official says Replicator initiative remains on track," U.S. Department of Defense, Jan. 26, 2024. [Online]. Available: https://www.defense.gov/News/News-Stories/Article/Article/3657609/defense-innovation-official-says-replicator-initiative-remains-on-track/. Accessed: Jun. 17, 2026.

[9] European Defence Agency, "EDA action plan on autonomous systems," Oct. 2024. [Online]. Available: https://eda.europa.eu/publications-and-data/publications/eda-action-plan-on-autonomous-systems. Accessed: Sep. 22, 2026.

[10] State Council of the People's Republic of China, "New generation artificial intelligence development plan," Jul. 2017, in Chinese. [Online]. Available: https://app.www.gov.cn/govdata/gov/201707/20/408540/article.html. Accessed: Sep. 22, 2026.

[11] Ministry of Industry and Information Technology of the People's Republic of China, "Guiding opinions on the innovative development of humanoid robots," Oct. 2023, in Chinese. [Online]. Available: https://www.ncsti.gov.cn/zcfg/zcwj/202311/t20231103_140346.html. Accessed: Jun. 17, 2026.

[12] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-physical systems security—A survey," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1802–1831, 2017, doi: 10.1109/JIOT.2017.2703172.

[13] G. Bakirtzis, B. Carter, C. H. Fleming, and C. R. Elks, "Mission aware cyber-physical security," *Syst. Eng.*, vol. 29, no. 2, pp. 354–367, 2026, doi: 10.1002/sys.70018.

[14] M. N. Slusher, "Lessons from the Ukraine conflict: Modern warfare in the age of autonomy, information, and resilience," Center for Strategic and International Studies, May 2025. [Online]. Available: https://www.csis.org/analysis/lessons-ukraine-conflict-modern-warfare-age-autonomy-information-and-resilience. Accessed: Jun. 17, 2026.

[15] Viasat, Inc., "KA-SAT network cyber attack overview," Mar. 2022. [Online]. Available: https://www.viasat.com/perspectives/corporate/2022/ka-sat-network-cyber-attack-overview/. Accessed: Jul. 19, 2026.

[16] F. Bajak, "Satellite modems nexus of worst cyberattack of Ukraine war," *Associated Press*, Mar. 30, 2022. [Online]. Available: https://apnews.com/article/russia-ukraine-technology-business-europe-broadband-internet-895f8aad2e71f56a5930aeaf833ff20f. Accessed: Jul. 26, 2026.

[17] J. Pearson, "Russia downed satellite internet in Ukraine—Western officials," *Reuters*, May 10, 2022. [Online]. Available: https://www.reuters.com/world/europe/russia-behind-cyberattack-against-satellite-internet-modems-ukraine-eu-2022-05-10/. Accessed: Jul. 26, 2026.

[18] Reuters, "Russia tries to jam Musk's Starlink systems to counter Ukrainian drones," Jul. 2026. [Online]. Available: https://www.reuters.com/business/aerospace-defense/russia-tries-jam-musks-starlink-systems-counter-ukrainian-drones-2026-07-08/. Accessed: Jul. 12, 2026.

[19] F. Casaril and L. Galletta, "Securing SatCom user segment: A study on cybersecurity challenges in view of IRIS^2," *Comput. Secur.*, vol. 140, Art. no. 103799, 2024, doi: 10.1016/j.cose.2024.103799.

[20] M. R. Asghar, Q. Hu, and S. Zeadally, "Cybersecurity in industrial control systems: Issues, technologies, and challenges," *Comput. Netw.*, vol. 165, Art. no. 106946, 2019, doi: 10.1016/j.comnet.2019.106946.

[21] J. Shen, J. Y. Won, Z. Chen, and Q. A. Chen, "Drift with devil: Security of multi-sensor fusion-based localization in high-level autonomous driving under GPS spoofing," in *Proc. 29th USENIX Security Symp.*, 2020, pp. 931–948.

[22] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao, "Adversarial sensor attack on LiDAR-based perception in autonomous driving," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2019, pp. 2267–2281, doi: 10.1145/3319535.3339815.

[23] J. Sun, Y. Cao, Q. A. Chen, and Z. M. Mao, "Towards robust LiDAR-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures," in *Proc. 29th USENIX Security Symp.*, 2020, pp. 877–894.

[24] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "DolphinAttack: Inaudible voice commands," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2017, pp. 103–117, doi: 10.1145/3133956.3134052.

[25] T. Sugawara, B. Cyr, S. Rampazzi, D. Genkin, and K. Fu, "Light commands: Laser-based audio injection attacks on voice-controllable systems," in *Proc. 29th USENIX Security Symp.*, 2020, pp. 2631–2648.

[26] X. Yuan, Y. Chen, Y. Zhao, Y. Long, X. Liu, K. Chen, S. Zhang, H. Huang, X. Wang, and C. A. Gunter, "CommanderSong: A systematic approach for practical adversarial voice recognition," in *Proc. 27th USENIX Security Symp.*, 2018, pp. 49–64.

[27] S. R. Hussain, O. Chowdhury, S. Mehnaz, and E. Bertino, "LTEInspector: A systematic approach for adversarial testing of 4G

LTE," in *Proc. Netw. Distrib. Syst. Security Symp.*, 2018, doi: 10.14722/ndss.2018.23313.

[28] D. Rupprecht, K. Kohls, T. Holz, and C. Pöpper, "IMP4GT: Impersonation attacks in 4G networks," in *Proc. Netw. Distrib. Syst. Security Symp.*, 2020, doi: 10.14722/ndss.2020.24283.

[29] H. Yang, S. Bae, M. Son, H. Kim, S. M. Kim, and Y. Kim, "Hiding in plain signal: Physical signal overshadowing attack on LTE," in *Proc. 28th USENIX Security Symp.*, 2019, pp. 55–72.

[30] D. Quarta, M. Pogliani, M. Polino, F. Maggi, A. M. Zanchettin, and S. Zanero, "An experimental security analysis of an industrial robot controller," in *Proc. IEEE Symp. Security Privacy*, 2017, pp. 268–286, doi: 10.1109/SP.2017.20.

[31] K. Chung, X. Li, P. Tang, Z. Zhu, Z. T. Kalbarczyk, R. K. Iyer, and T. Kesavadas, "Smart malware that uses leaked control data of robotic applications: The case of Raven-II surgical robots," in *Proc. Int. Symp. Res. Attacks, Intrusions Defenses*, 2019, pp. 337–351.

[32] A. T. Markettos, C. Rothwell, B. F. Gutstein, A. Pearce, P. G. Neumann, S. W. Moore, and R. N. M. Watson, "Thunderclap: Exploring vulnerabilities in operating system IOMMU protection via DMA from untrustworthy peripherals," in *Proc. Netw. Distrib. Syst. Security Symp.*, 2019, doi: 10.14722/ndss.2019.23194.

[33] A. Tang, S. Sethumadhavan, and S. J. Stolfo, "CLKSCREW: Exposing the perils of security-oblivious energy management," in *Proc. 26th USENIX Security Symp.*, 2017, pp. 1057–1074.

[34] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," in *Proc. Netw. Distrib. Syst. Security Symp.*, 2018, doi: 10.14722/ndss.2018.23291.

[35] E. Bagdasaryan and V. Shmatikov, "Blind backdoors in deep learning models," in *Proc. 30th USENIX Security Symp.*, 2021, pp. 1505–1521.

[36] D. Kumar, R. Paccagnella, P. Murley, E. Hennenfent, J. Mason, A. Bates, and M. Bailey, "Skill squatting attacks on Amazon Alexa," in *Proc. 27th USENIX Security Symp.*, 2018, pp. 33–47.

[37] W. Ding, S. Liao, L. Cheng, X. Mi, Z. Zhao, and H. Hu, "Command hijacking on voice-controlled IoT in Amazon Alexa platform," in *Proc. ACM Asia Conf. Comput. Commun. Security*, 2024, pp. 654–666, doi: 10.1145/3634737.3657010.

[38] N. Zhang, X. Mi, X. Feng, X. Wang, Y. Tian, and F. Qian, "Dangerous skills: Understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems," in *Proc. IEEE Symp. Security Privacy*, 2019, pp. 1381–1396.

[39] H. Fu, S. Huang, Z. Zhang, N. Xi, Y. Shen, X. Zhu, and F. Li, "ATCAuth: A P2P device authentication method based on WLAN access temporal characteristics," in *Proc. 2025 Int. Conf. Networking and Network Applications (NaNA)*, 2025, pp. 17–23, doi: 10.1109/NaNA66698.2025.00011.

[40] R. Zhou, Z. Zhang, H. Han, X. Zhang, Z. Chen, J. Sun, Y. Shen, and D. Xu, "RSFuzz: A robustness-guided swarm fuzzing framework based on behavioral constraints," in *Proc. 40th IEEE/ACM Int. Conf. Automated Software Engineering (ASE)*, 2025, pp. 495–507, doi: 10.1109/ASE63991.2025.00048.

[41] Y. Huang, Z. Xiong, and Q. Zhu, "Cross-layer coordinated attacks on cyber-physical systems: A LQG game framework with controlled observations," in *Proc. Eur. Control Conf.*, 2021, pp. 521–528, doi: 10.23919/ECC54610.2021.9654874.

[42] J. Tian, B. Wang, T. Li, F. Shang, and K. Cao, "Coordinated cyber-physical attacks considering DoS attacks in power systems," *Int. J. Robust Nonlinear Control*, vol. 30, no. 11, pp. 4345–4358, Jul. 2020, doi: 10.1002/rnc.4801.

[43] Z. Li, M. Shahidehpour, A. Alabdulwahab, and A. Abusorrah, "Bilevel model for analyzing coordinated cyber-physical attacks on power systems," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2260–2272, Sep. 2016, doi: 10.1109/TSG.2015.2456107.

[44] R. Deng, P. Zhuang, and H. Liang, "CCPA: Coordinated cyber-physical attacks and countermeasures in smart grid," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2420–2430, Sep. 2017, doi: 10.1109/TSG.2017.2702125.

[45] Y. Chen, S. Lakshminarayana, and F. Teng, "Localization of coordinated cyber-physical attacks in power grids using moving target defense and deep learning," in *Proc. IEEE Int. Conf. Commun., Control, Comput. Technol. Smart Grids*, 2022, pp. 387–392, doi: 10.1109/SmartGridComm52983.2022.9960982.

[46] S. Lakshminarayana, E. V. Belmega, and H. V. Poor, "Moving-target defense against cyber-physical attacks in power grids via game theory," *IEEE Trans. Smart Grid*, vol. 12, no. 6, pp. 5244–5257, Nov. 2021, doi: 10.1109/TSG.2021.3095083.

[47] Y. Huang, T. He, N. R. Chaudhuri, and T. F. La Porta, "Preventing outages under coordinated cyber-physical attack with secured PMUs," *IEEE Trans. Smart Grid*, vol. 13, no. 4, pp. 3160–3173, Jul. 2022, doi: 10.1109/TSG.2022.3165768.

[48] J. A. Guerrero-Saade and M. van Amerongen, "AcidRain: A modem wiper rains down on Europe," SentinelLabs, Mar. 2022. [Online]. Available: https://www.sentinelone.com/labs/acidrain-a-modem-wiper-rains-down-on-europe/. Accessed: Jul. 19, 2026.

[49] ESET Research, "Industroyer2: Industroyer reloaded," Apr. 2022. [Online]. Available: https://www.welivesecurity.com/2022/04/12/industroyer2-industroyer-reloaded/. Accessed: Jul. 19, 2026.

[50] F. Bajak, "Ukraine says potent Russian hack against power grid thwarted," *Associated Press*, Apr. 12, 2022. [Online]. Available: https://apnews.com/article/russia-ukraine-kyiv-technology-business-hacking-0147e33bc1846a3f8039f9c65a1b4b50. Accessed: Jul. 26, 2026.

[51] J. Pearson, "Ukraine says it thwarted Russian cyberattack on electricity grid," *Reuters*, Apr. 12, 2022. [Online]. Available: https://www.reuters.com/world/europe/russian-hackers-tried-sabotage-ukrainian-power-grid-officials-researchers-2022-04-12/. Accessed: Jul. 26, 2026.

[52] T. H. Sakib, Y. R. Martinez, C. Brady, S. R. Hasan, and T. N. Guo, "Supply chain exploitation of secure ROS 2 systems: A proof-of-concept on autonomous platform compromise via keystore exfiltration," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, 2025, pp. 1–6, doi: 10.1109/MILCOM64451.2025.11309897.

[53] Z. Guo, Z. Lin, P. Li, and K. Chen, "SkillExplorer: Understanding the behavior of skills in large scale," in *Proc. 29th USENIX Security Symp.*, 2020, pp. 2649–2666.

[54] Mandiant, "TRITON malware: Attackers deploy new ICS attack framework," 2017. [Online]. Available: https://cloud.google.com/blog/topics/threat-intelligence/attackers-deploy-new-ics-attack-framework-triton. Accessed: Jul. 12, 2026.

[55] Cybersecurity and Infrastructure Security Agency, "Advanced persistent threat compromise of government agencies, critical infrastructure, and private sector organizations," Cybersecurity Advisory AA20-352A, Dec. 17, 2020, updated Apr. 15, 2021. [Online]. Available: https://www.cisa.gov/news-events/cybersecurity-advisories/aa20-352a. Accessed: Jul. 12, 2026.

[56] Mandiant, "3CX software supply chain compromise initiated by a prior software supply chain compromise; suspected North Korean actor responsible," 2023. [Online]. Available: https://cloud.google.com/blog/topics/threat-intelligence/3cx-software-supply-chain-compromise. Accessed: Jul. 12, 2026.

[57] A. Greenberg, "The untold story of NotPetya, the most devastating cyberattack in history," *Wired*, 2018. [Online]. Available: https://www.wired.com/story/notpetya-cyberattack-ukraine-russia-code-crashed-the-world/. Accessed: Jul. 12, 2026.

[58] Microsoft Defender Security Research Team, "New ransomware, old techniques: Petya adds worm capabilities," Jun. 27, 2017. [Online]. Available: https://www.microsoft.com/en-us/security/blog/?p=75612. Accessed: Sep. 21, 2026.

[59] Cisco Talos, "New VPNFilter malware targets at least 500K networking devices worldwide," 2018. [Online]. Available: https://blog.talosintelligence.com/vpnfilter/. Accessed: Jul. 12, 2026.

[60] National Cyber Security Centre, "New Sandworm malware Cyclops Blink replaces VPNFilter," Joint Advisory, 2022. [Online]. Available: https://www.ncsc.gov.uk/news/new-sandworm-malware-cyclops-blink-replaces-vpnfilter. Accessed: Jul. 12, 2026.

[61] Cybersecurity and Infrastructure Security Agency, "PRC state-sponsored actors compromise and maintain persistent access to U.S. critical infrastructure," Cybersecurity Advisory AA24-038A, 2024. [Online]. Available: https://www.cisa.gov/news-events/cybersecurity-advisories/aa24-038a. Accessed: Jul. 12, 2026.

[62] C. Pascoe, S. Quinn, and K. Scarfone, "The NIST cybersecurity framework (CSF) 2.0," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Cybersecurity White Paper 29, Feb. 2024, doi: 10.6028/NIST.CSWP.29.

[63] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 800-207, Aug. 2020, doi: 10.6028/NIST.SP.800-207.

[64] Z. Xu, Z. Zhang, G. Tang, Z. Shi, Y. Shen, and N. Xi, "Multidimensional intrinsic identity construction and dynamic seamless authentication schemes in IoT environments," in *Algorithms and Architectures for Parallel Processing*, Lecture Notes in Computer Science, vol. 15254. Singapore: Springer, 2025, pp. 134–153, doi: 10.1007/978-981-96-1545-2_9.

[65] G. Tang, Z. Zhang, and N. Xi, "SPUAuth: Strengthening pattern unlock authentication with behavioral biometrics against shoulder surfing attacks," in *Mobile and Ubiquitous Systems: Computing, Networking*

and Services, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 683. Cham, Switzerland: Springer, 2026, pp. 335–355, doi: 10.1007/978-3-032-22500-9_19.

[66] Z. Xu, Z. Zhang, G. Tang, Y. Chen, and Y. Wang, "An anomaly causal analysis method in complex 5G network environments," in *Broadband Communications, Networks, and Systems*, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 674. Cham, Switzerland: Springer, 2026, pp. 177–182, doi: 10.1007/978-3-032-14350-1_13.

[67] H. Han, B. Zha, K. Wang, Z. Zhang, Y. Shen, J. Wang, Y. Kang, and X. Zhu, "A semantic provenance graph-based detection method for ransomware attacks," in *Proc. 2025 Int. Conf. Networking and Network Applications (NaNA)*, 2025, pp. 296–302, doi: 10.1109/NaNA66698.2025.00054.

[68] J. Boyens, A. Smith, N. Bartol, K. Winkler, A. Holbrook, and M. Fallon, "Cybersecurity supply-chain risk-management practices for systems and organizations," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 800-161 Revision 1, Update 1, Nov. 2024, doi: 10.6028/NIST.SP.800-161r1-upd1.

[69] M. Souppaya, K. Scarfone, and D. Dodson, "Secure software development framework (SSDF) version 1.1: Recommendations for mitigating the risk of software vulnerabilities," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 800-218, Feb. 2022, doi: 10.6028/NIST.SP.800-218.

[70] R. Ross, V. Pillitteri, R. Graubart, D. Bodeau, and R. McQuaid, "Developing cyber-resilient systems: A systems-security-engineering approach," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 800-160 Volume 2 Revision 1, Dec. 2021, doi: 10.6028/NIST.SP.800-160v2r1.

[71] E. Tabassi, "Artificial intelligence risk management framework (AI RMF 1.0)," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST AI 100-1, 2023, doi: 10.6028/NIST.AI.100-1.

[72] A. Vassilev, A. Oprea, A. Fordyce, and H. Anderson, "Adversarial machine learning: A taxonomy and terminology of attacks and mitigations," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST AI 100-2e2023, 2024, doi: 10.6028/NIST.AI.100-2e2023.

[73] International Organization for Standardization, "Information technology—Artificial intelligence—Guidance on risk management," ISO/IEC 23894:2023, Geneva, Switzerland, 2023.

[74] International Organization for Standardization, "Information technology—Artificial intelligence—Management system," ISO/IEC 42001:2023, Geneva, Switzerland, 2023.

[75] R. Ross, M. Winstead, and M. McEvilley, "Engineering trustworthy secure systems," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 800-160 Volume 1 Revision 1, 2022, doi: 10.6028/NIST.SP.800-160v1r1.

[76] European Parliament and Council of the European Union, "Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence," *Official Journal of the European Union*, 2024. [Online]. Available: http://data.europa.eu/eli/reg/2024/1689/oj. Accessed: Jul. 14, 2026.