The FAA's Center of Excellence for UAS Research

ASSURE

Alliance for System Safety of UAS through Research Excellence

Appendix A:

Illustrate the Need for UAS Cybersecurity Oversight & Risk Management: Literature Review

March 20, 2024

NOTICE

LEGAL DISCLAIMER

The information provided herein may include content supplied by third parties. Although the data and information contained herein has been produced or processed from sources believed to be reliable, the Federal Aviation Administration makes no warranty, expressed or implied, regarding the accuracy, adequacy, completeness, legality, reliability or usefulness of any information, conclusions or recommendations provided herein. Distribution of the information contained herein does not constitute an endorsement or warranty of the data or information provided herein by the Federal Aviation Administration or the U.S. Department of Transportation. Neither the Federal Aviation Administration nor the U.S. Department of Transportation shall be held liable for any improper or incorrect use of the information contained herein and assumes no responsibility for anyone's use of the information. The Federal Aviation Administration and U.S. Department of Transportation shall not be liable for any claim for any loss, harm, or other damages arising from access to or use of data or information, including without limitation any direct, indirect, incidental, exemplary, special or consequential damages, even if advised of the possibility of such damages. The Federal Aviation Administration shall not be liable to anyone for any decision made or action taken, or not taken, in reliance on the information contained herein.

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. A11L.UAS.95_A58	**2. Government Accession No.**	**3. Recipient's Catalog No.**
4. Title and Subtitle Illustrate the Need for UAS Cybersecurity Oversight & Risk Management: Literature Review		**5. Report Date** January 15, 2024 **6. Performing Organization Code**
7. Author(s) Perry Alexander, PhD https://orcid.org/0000-0002-5387-9157 Adam Petz, PhD Steven Weber, PhD Rakesh Bobba, PhD		**8. Performing Organization Report No.**
9. Performing Organization Name and Address I2S - The University of Kansas, 2335 Irving Hill Rd, Lawrence, KS Drexel University, 3141 Chestnut St, Philadelphia, PA 19104 Oregon State University, 1500 SW Jefferson Way, Corvallis, OR 97331		**10. Work Unit No.** **11. Contract or Grant No.**
12. Sponsoring Agency Name and Address Federal Aviation Administration		**13. Type of Report and Period Covered** Literature Survey **14. Sponsoring Agency Code** 5401

15. Supplementary Notes

16. Abstract

There is no established framework for addressing cybersecurity in UASs. This literature survey outlines the GAO and NIST reports on cybersecurity frameworks and asserts that while necessary, they are not sufficient for protecting airspace from cybersecurity incidents. The presence of cyberphysical interfaces dominates the UAS attack surface and yet is not specifically identified in general purpose frameworks. We provide a malware literature survey identifying the various issues that must be addressed in a UAS framework. We propose identifying issues related to highly-likely, high-risk attacks in the A38 literature survey and developing a framework that addresses those issues. The result will be a cybersecurity framework that addresses traditional security issues while focusing cyberphysical attacks common in UASs.

17. Key Words UAS, cybersecurity framework, cyberphysical systems	**18. Distribution Statement** No restrictions. This document is available through the National Technical Information Service, Springfield, VA 22161.

19. Security Classification (of this report) Unclassified	**20. Security Classification (of this page)** Unclassified	**21. No. of Pages** 39	**22. Price**

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

TABLE OF CONTENTS

TABLE OF ACRONYMS

AI	Artificial Intelligence
ANN	Artificial Neural Network
AP	Access Point
CVE	Common Vulnerabilities and Exposures
DDoS	Distributed Denial of Service
DoS	Denial of Service
FAA	Federal Aviation Administration
GAO	Government Accounting Office
GCS	Ground Control Station
GPS	Global Positioning System
IDS	Intrusion Detection System
IFR	Instrument Rules
ILS	Instrument Landing System
IoT	Internet of Things
IoTBOX	IoT Sandbox
IoTPOT	IoT Honeypot
MEMS	Micro-Electro-Mechanical Systems
NAVAID	Navigational Aid
NAS	National Airspace System
NIST	National Institute of Standards and Technology
OS	Operating System
RNN	Recurrent Neural Network
ROP	Return Oriented Programming
TCP	Transmission Control Protocol
UAS	Unpiloted Aircraft Systems
UAV	Unpiloted Aerial Vehicles

EXECUTIVE SUMMARY

The ever-growing presence of Unpiloted Aircraft Systems (UAS) in the national airspace increases the need to address cybersecurity issues in UAS infrastructure. Adversaries that gain control of or interfere with UAS in the airspace can damage that infrastructure in ways ranging from interference with missions to launching attacks. As of today, there is no commonly accepted framework for addressing UAS-specific cybersecurity issues. While general frameworks exist for cybersecurity broadly, the nature of airspace and UAS issues necessitate a specialized UAS approach. The objectives in this project establishing the need for a UAS cybersecurity framework and provide a prototype framework. The objective in this report is gathering information about the state-of-the-practice in UAS cybersecurity and establish the needs for a UAS specific framework.

The approach is to begin with the Government Accounting Office (GAO)-19-105 report and National Institute of Standards and Technology (NIST) Framework for Improving Critical Infrastructure Cybersecurity frameworks to determine if they are sufficient. The researchers then overview types of malware potentially present in UAS to establish that the cyberphysical nature of UAS makes them vulnerable to new classes of malware. The conclusion has an overview of how the team will produce a UAS cybersecurity framework in the context of the findings.

The GAO and NIST reports are not sufficient for use in the UAS domain. GAO-19-105 is not a framework, but a report on the application of frameworks across government agencies. It argues for improved oversight suggesting that current cybersecurity practices and policies lack proper enforcement. As such, the GAO report supports the conclusion that UAS need an enforceable framework while providing no direction on what that framework should be. The NIST Framework for Improving Critical Infrastructure Cybersecurity is a general framework for communicating, enforcing, and evaluating cybersecurity infrastructure. The NIST framework defines five activities – Identify, Protect, Detect, Respond, and Recover – that specify the stages of intervention. It continues by defining four readiness tiers useful in evaluating an organization's readiness. Tiers range from partial, *ad hoc* approaches to adaptive approaches that respond and adapt to evolving cybersecurity situations.

Following the literature review, experimentation, and discussion the researchers determined the NIST framework is *necessary for UASs but is not sufficient*. This report documents various malware classes that apply to UAS systems. From that effort it is clear that UAS systems are predominantly cyberphysical in nature. A significant interface exists between the UAS and the physical world it occupies. The presence of sensors and actuators at this interface presents a broad attack surface that is not present in general computing systems. Spoofing and compromising sensors to manipulate actuators and have physical impact represents a significant attack surface that must be addressed in UAS design.

Building upon extensive documentation from the A38 UAS Cyber Security and Safety Literature Review, the researchers will identify attacks that are likely to impact airspace. Rather than focus on more traditional information-centered attacks, the work will focus on cyberphysical compromises with impacts on airspace safety and security.

1 INTRODUCTION & BACKGROUND

There is no framework for addressing cybersecurity issues specific to fielding Uncrewed or Optionally Piloted Aircraft Systems (UAS). The objective of this survey is to review general systems and UAS specific cybersecurity-related frameworks. It starts with an overview of general frameworks including GAO-19-105 and the NIST Framework for Improving Critical Infrastructure Cybersecurity. It then explores issues in embedded systems generally and UAS specifically identifying why a targeted approach is required for UAS cybersecurity. Finally, the report builds from the ASSURE A38 UAS Cyber Security and Safety Literature Review, using their attack likelihood and severity classification to identify critical to airspace protection.

2 GENERAL CYBERSECURITY FRAMEWORKS

2.1 GAO-19-105

The GAO-19-105 report provides insight into government implementation of recommended cybersecurity practices. The conclusion of the report is many government agencies neglect to implement and maintain NIST recommended cybersecurity defenses. Capabilities recommended for implementation by NIST include: (i) monitoring cloud services; (ii) using intrusion prevention systems; (iii) monitoring both external and internal network traffic; (iv) and implementing a system for managing security information and security events. GAO-19-105 reports that implementation is inconsistent at best with no action taking on NIST recommendations. Agencies interviewed reported they had failed to implement one or more of these capabilities. Summarizing conclusions:

- Less than half of agencies using cloud-based infrastructure monitored inbound and outbound network traffic.
- Less than half of agencies used host-based intrusion prevention capabilities although most used some kind of intrusion prevention system.
- Less than half of agencies persistently monitored encrypted traffic, and several were not monitoring traffic at all.
- Logging is performed inconsistently across agencies, with only five collecting all logs for matching with known vulnerabilities and threats.

GAO-19-105 makes a collection of recommendations to the Department of Homeland Security and Office of Management and Budget. While largely administrative in nature, the recommendations provide a path for policy development and enforcement.

While GAO-19-105 provides insight into the cybersecurity problem important for policymakers and leadership, it provides little insight into how government agencies might implement and maintain these services. However, it does reference the NIST cybersecurity framework developed around five functions: (i) identify; (ii) protect; (iii) detect; (iv) respond; and (v) recover. These functions and the associated framework will be discussed in the subsequent section.

The research team's conclusions about GAO-19-105 are that while it is a useful report on the state-of-the-practice for government agencies with respect to NIST recommendations, it provides no

guidance for solving related problems beyond several high-level agency recommendations. Even then, the report provides no guidance specific to aircraft – crewed or uncrewed – or to general cyberphysical issues critical to their protection. It is the researchers' recommendation that GAO-19-105 be revisited and methodologies applied to aviation and cyberphysical systems before its conclusions will contribute to UAS security.

2.2 NIST Framework for Improving Critical Infrastructure Cybersecurity

The NIST Cybersecurity Framework v2.0 is a general-purpose framework for protecting systems from cyberattack. Like GAO-19-105, its target is general computing systems and not aerospace systems, but it provides useful guidance for addressing cybersecurity risks.

The NIST Framework provides a common structure for discussing cybersecurity mitigation. At its heart are five Framework Core Functions that should be continuously performed to address cybersecurity risks. The five Core Functions are:

- Identify – Develop a cross-cutting organizational approach to managing cybersecurity risk to systems, people, assets, data and capabilities.
- Protect – Develop and implement safeguards that protect availability of critical services.
- Detect – Develop and implement means for detecting the presence of a cybersecurity threat.
- Respond – Develop and implement actionable responses to a detected cybersecurity issue.
- Recover – Develop and implement resilient systems that maintain and restore services under attack.

These core functions are quite general and applicable to virtually any system from a server farm to a home automation Internet of Things (IoT) device. They define a path from preparation through detection and mitigation of a cyberattack. The key takeaway is the Core Functions define a vocabulary for discussion and a taxonomy for the various actions taken to ensure system availability.

Next, the framework defines Implementation Tiers that document how an organization should view cybersecurity risks and management of those risks. Implementation Tiers are descriptive of an organization's preparedness and awareness of cybersecurity risks. All tiers are described in terms of three concepts:

- Risk Management Process
- Integrated Risk Management Program
- External Participation

The Partial Tier (Tier 1) uses an *ad hoc* approach to risk management that is not formally defined and is typically reactive. There is limited awareness of cybersecurity risk and risk management is performed in non-systematic ways on a case-by-case basis. A Tier 1 organization typically does not understand its role in its parent organization or in relation to other peer organizations. Partial Tier organizations are best described as not having any systematic approach to cybersecurity in the small or in the large.

The Risk Informed Tier (Tier 2) uses organizationally approved practices that may not be implemented or implemented unevenly across the organization. The organization is aware of cybersecurity risks, but management is not performed at the organization level. A Tier 2 organization may understand its role in a larger organization, but only partially.

The Repeatable Tier (Tier 3) builds upon Tier 2 adding a repeatable criterion for cybersecurity practices. Policies and practices are formally defined and approved with regular updates by the organization. An organization-wide approach to cybersecurity risk management exists with methods in place to respond to changes in risks. Monitoring is consistent and accurate with the Tier 3 organization understanding its role as well as its dependencies, dependents, and operational assumptions. The key description is repeatable where all functions are defined and approved.

The Adaptive Tier (Tier 4) is repeatable with additional capabilities for adaptation in response to new information and new adversary capabilities. There is an organizational approach to cybersecurity that uses risk-informed policies, processes, and procedures to address attacks. The key is a connection between risks introduced by cyberattacks and organizational objectives with objectives guiding responses to specific events. The Tier 4 organization understands its place in the larger organization with knowledge of dependents, dependencies, and working assumptions. Further the organization is in continuous contact, exchanging information to implement highly informed, adaptive processes.

The NIST standard claims that Tiers are not capability measures. However, one clearly wants to move their organization from Tier 1 to higher tiers to demonstrate increasing maturity. Moving from *ad hoc* processes that are not repeatable or approved to an organization where cybersecurity risks are first-class in the organization, continuously approached, and continuously improved is desired.

Finally, the NIST framework defines a Profile that describes an implementation of the framework in a specific system. Profiles are the objects that are assessed and compared in the various framework components. A Profile should support rigorous analysis and comparison as well as providing a portable framework implementation.

The research team's conclusions about the NIST Cybersecurity Framework are that it is a useful guide for organizing and deploying cybersecurity assets. It is widely accepted as a standard for securing systems generally and in that respect applies equally well to UAS. However, like the GAO-19-105 report, it does not specifically address cyberphysical issues that cannot be ignored in systems that interact with their environments. There is no question the NIST Framework can be applied to UAS, but it provides no out-of-the-box guidance for deploying systems. It is minimally required that the researchers develop framework profiles for UAS. More appropriately, the NIST Framework should be revised to address UAS specific issues.

2.3 Industry Discussions

To better understand the UAS cybersecurity environment we engaged several members of the KU Science of Security industry advisory board. This interaction was informal, and we agreed to hold

the companies and identities of participants confidential to avoid legal and permission issues. That said, our three participants represent two major aerospace equipment providers and one major telecommunications company. Between them they provide telecommunications and flight systems for both government and commercial customers. They have a combined 70 years experience developing systems ranging from commercial airliners to small aircraft to cutting edge UASs. All have extensive experience in cybersecurity as practitioners and researchers.

All participants agreed on the need for guiding principles and frameworks for UAS cybersecurity and that no such frameworks exist. Our industry participants all encouraged active participation in existing standards efforts. One individual who participates in severl international cybersecurity standards organizations discussed efforts to standardize autonomous flight control systems generally. They recommended several documents central to their efforts:

- RTCA DO-356A/EUROCAE ED-203A - Airworthiness Security Methods and Considerations.
- RTCA DO-377A - UAS Command and Control Minimum Aviation System Performance Standards.
- FAA Order 1370.121B - FAA Information Security and Privacy Program and Policy.

Note that these were also identified by our FAA advisors. Interestingly, the RCTA DO-356A document is at the heart of several existing certification processes. A quick online survey reveals AdaCore, LDRA Assured, Mayhem Security as example companies actively involved with RCTA DO-356A compliance. Certification is ongoing in aerospace systems, but without specific guidance for UASs.

2.3.1 Legal, Technical, Policy

Our industry advisors all agreed on the importance of an integrated approach to UAS cybersecurity. Specifically, continuing the development of a technical framework for cybersecurity while working with standards bodies to develop policy and the legal community to develop enforcement mechanisms. Issues with new technologies cannot be solved by technical solutions alone.

The community must develop policies that describe best practices for integrating cybersecurity in workflows. Much like traditional software engineering, these best practices should be taught and implemented around the technical development process. We were encouraged to include policy issues as a part of our framework proposal for this effort which should occur naturally as we proceed.

Developing a legal framework is not something our industry partners or we have expertise in. We did discuss working with legal experts in the future, but no specific details. One representative emphasized that policy and law are quite distinct things that must be developed separately.

Technical solutions are where we excel as engineers, thus our framework will focus largely on technical issues. However, one of our industry advisors emphasized that we should be developing policy and practices along with the framework or as a part of the framework.

2.3.2 Cyberphysical Systems

One primary conclusion from our initial research is that cyberphysical issues often define UAS cybersecurity. The interface between the UAS and the physical world through sensors and actuators is a critical part of the UAS adversary model. All our advisers agreed with this assessment, but one emphasized that we cannot ignore information exfiltration.

Our focus is on protecting the airspace implying protecting UAS control systems. We bracket out information exfiltration in the traditional sense. One of our advisors points out that security is a system-level problem, making it impossible to separate information exfiltration and attacks on UAS controls. They use the example of a map database to illustrate their point. A traditional cyberattack might attempt to steal map information from the UAS or spoof map data. The impacts of such an attack are realized in the map database, but also have serious impact on control of the aircraft. While the map database is not a cyberphysical issue, it has impact on the behavior of the UAS potentially impacting the airspace. When developing our framework, we must include address cybersecurity in the broadest sense while focusing on cyberphysical systems.

2.3.3 Legacy Systems

One of our advisors brought up the issue of legacy systems in flight control software and suggested that should be considered in our framework. Rarely do we get to start from scratch, particularly in highly regulated and certified software like UAS systems specifically or aerospace system generally.

We have not called out legacy systems in our investigation as a separate issue. However, we have performed all our demonstrations on legacy software. UxAS and Ardupilot being our most popular demonstration systems. While we agreed that legacy systems are an important consideration, we are dealing with legacy systems as a natural part of our investigation. Unless directed otherwise we will continue our current trajectory that does not call out legacy systems as a separate issue.

2.3.4 Limitations

Of all our tasks, the industry study was most impacted by Covid. We had a great start interacting with our board, but the shutdown ended most of our interactions. Specifically, our Science of Security board stopped meeting regularly limiting our opportunities to interact with them. The board is reforming; thus we hope to include them in a final review of our framework.

3 UAS & EMBEDDED SYSTEMS ISSUES

In this section, the researchers survey and categorize the security threats that UAS and cyber-physical embedded systems face, leveraging among other efforts the literature survey from the ASSURE A38 project. Because of their interaction with the environment, UAS and cyber-physical systems can have unique cyber-security challenges compared to traditional IT environments.

As discussed previously, the NIST Cybersecurity Framework v2 provides a process for managing organizational cybersecurity risks organized into the following core strategies or functions:

Govern, Identify, Protect, Detect, Respond, and Recover. Each core function is further broken-down into categories and sub-categories that describe a desired outcome. Organizations or teams looking to leverage this framework need to decide how to achieve those desired outcomes which involves balancing potential cybersecurity risks faced with costs of mitigating those risks.

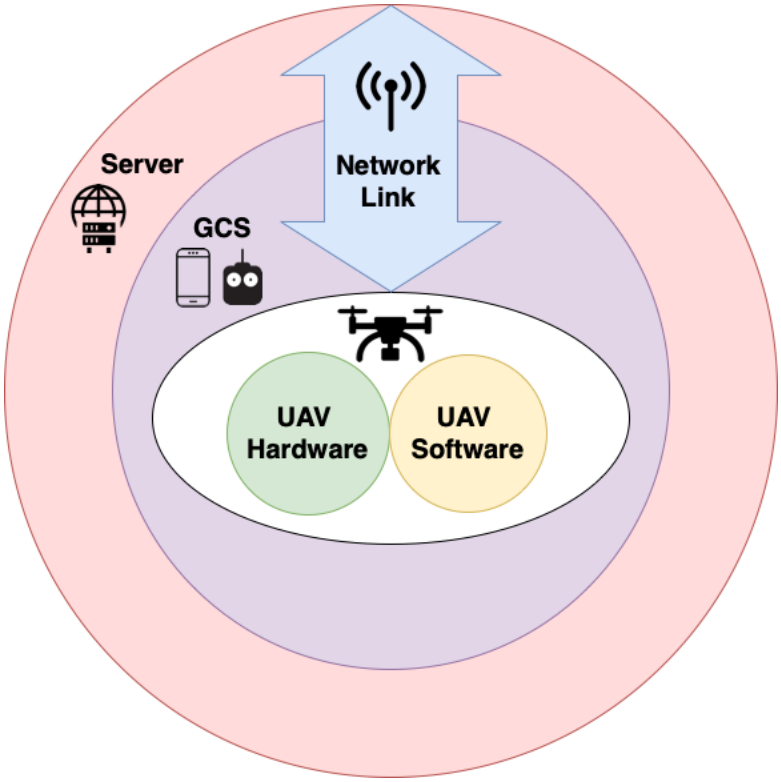

Figure 1. Components of UAS (from A38).

- **UAS Hardware**: The hardware components of a UAS include physical components of the UAS, such as body, propellers, sensors (e.g., Global Positioning System [GPS], Inertial Measurement Units, camera), actuators (e.g., motor).
- **UAS Software**: Software components of UAS include firmware, operating system, application programs, and control algorithm implementations.
- **Ground Control Station (GCS)**: The system that controls or communicates with the UAS remotely from the ground. It may consist of hardware and software of the remote site/controller and the operator (e.g., human pilot).
- **Network/Communication Link**: This includes communication networks and channels, and protocols used in UAS. This mainly refers to drone-to-GCS communication, but it also

includes drone-to-drone, drone/GCS-to-server communications, and communications with services such as GPS. Communications may take place over Wi-Fi, ad-hoc networks, cellular, or other networks depending on the application and environment context.

- **Server/Cloud**: Server/Cloud refers to remotely located cloud servers or services that store information regarding UAS such as flight logs and registration information.

3.1 UAS Hardware Attacks

The two main categories of hardware attacks are jamming and spoofing, but they can be facilitated by attacks on supply chain or firmware. Jamming involves prevention of the communication of sensor data to the UAS controller. Spoofing is similar but achieves a higher level of control. It involves falsifying the sensor data or replaying valid signals that were previously captured. Knowing the characteristics and tolerances of the sensor components, or replacing them with less robust components, can enable a higher success rate of attacks (supply chain attack). Compromised firmware can also make it easier to alter the data from the sensor. Hardware components may include actuators, GPS receivers, inertial Micro-Electro-Mechanical Systems sensors (MEMS), various cameras, Wi-Fi receivers, light detection and ranging, or various general sensors like temperature.

Table 1. UAS Hardware Attack Descriptions.

Attack Method	**Description**
d	Various methods of creating false GPS signals or replaying valid ones to deceive UAS location
Spoofing - ADS-B	Sending false messages or modifying legitimate messages
Spoofing - Actuator	Sending electromagnetic noise to manipulate the signal going between the controller and actuator
Spoofing - Inertial MEMS Sensors	Acoustic signals or other methods can target resonance frequencies and alter sensor readings
Jamming - General	High energy noisy signals, large number of signals, and destructive interference signals using the signal medium of the target

	component can obscure the valid signals.
Firmware Flashing	Physically flashing the firmware to replace with malicious version
Supply Chain Attack	Modifying hardware components with faultier or less robust ones. Or knowing the component specifications to better target certain spoofing attacks (e.g. resonance frequency or noise)

3.2 UAS Software Attacks

This is a broad category that overlaps with others, but the focus is on operating systems, flight control algorithm implementation, and any application software run on the UAS. An attack can have multiple levels of outcomes depending on the situation, so focus isn't on the outcomes but rather the possible avenues for attack. Attacks on operating systems and application software are similar to that in other domains, however UAS typically run real-time operating systems with stricter timing requirements which creates some differences. These attacks can involve typical attacks such as buffer overflows and malware injection to enable an attacker to gain privileges and execute commands and control the system. However, attacks on control algorithms can differ significantly. Control algorithms rely on decision making with regards to the environment and sensor data. They have a unique attack surface where there may be relatively few lines of code, which are easier to secure, but large amounts of potential states, which are difficult to test. The attacker may try to push the system into awkward states that are not well handled by the control algorithm and may give the attacker some level of control over the UAS.

Table 2. UAS Software Attack Descriptions.

Attack Method	**Description**
General Exploits	Depending on the level of privilege, various traditional methods of bug exploits such as Return Oriented Programming (ROP) or buffer overflow can increase privilege, allow executing arbitrary commands or altering sensor data or wasting battery and processing power.
Firmware or Malware Modification	Software can be infected via physical ports (e.g. USB) or wireless channels. Related to supply chain attack.

Database Injection	If the UAS contains an accessible database, attacks such as SQL injection can insert malicious data or commands.
Algorithm attacks	Introducing unexpected combinations of states or events that aren't well captured by designed algorithms can disrupt or allow control over UAS. This can occur externally such as with interfering UAS or internally with injected messages or state alterations.
Adversarial Artificial Intelligence (AI)	Some UAS components may use AI such as computer vision for cameras. Adversarial attacks can trick AI into misclassification, for example in cameras the depth perception or obstacle detection. These attacks occur as a result of perception, but various methods such as supply chain can weaken the AI to make it more susceptible.
Supply Chain Attack	Inserting vulnerabilities into libraries or software development kits that UAS software may rely on. Gaining access to supplier computers and installing malware is also a possibility. Poisoning data used for modeling or training can weaken algorithms.

3.3 Ground Control Station (GCS) Attacks

GCS can have various forms – from a single human pilot with a remote device or smartphone, to a large ground facility with multiple operators that manage a fleet of drones. In some cases, the UAS may rely very little on the GCS. The researchers view the GCS as a standalone cyber-physical system, with its own hardware and software. GCS acts more like a traditional software system but may involve real time communication and human operators. Typical software, hardware, supply chain, and social engineering attacks may apply in order to compromise the system to enable the attacker to manipulate the drones and airspace it oversees. As with UAS software attacks, an attack can have multiple levels of outcomes depending on the situation and privilege level.

Table 3. GCS Attack Descriptions.

Attack Method	Description
General Exploits	Depending on the level of privilege, various traditional methods of bug exploits such as ROP or buffer overflow can increase privilege. For GCS this can enable attackers to launch attacks on UAS, control drones, or disrupt GCS software that has communication links to drones.
Data Exfiltration	Extracting information such as communication protocols, drone positions, drone specifications, and authentication can allow attackers to better target their attacks.
Reverse Engineering of GCS Software	Like data exfiltration but with a slightly different avenue, reverse engineering GCS software can find hardcoded authentication tokens or sensitive information
Social Engineering	Manipulation methods can get human operators to install malware, provide unauthorized access, or reveal sensitive information.
Supply chain	Like UAS software, software libraries or computers can be compromised to enable exploits.
Firmware or Malware Modification	Software can be infected via physical ports (e.g. USB) or wireless channels. Related to supply chain attack and social engineering.

3.4 Network/Communication Link Attacks

Network links are wireless communication channels used in UAS which can include commands and data used for UAS navigation and control. Networks contain software components but here this focus is more directly on the communication aspect since the software aspects are covered by the UAS and GCS software categories (e.g. jamming, spoofing). In this sense, consider network attacks as dependent on which data is being transmitted and which protocol is being used. These

attacks can be viewed as protocol and network topology aware spoofing. Compromised networks can cause the UAS to receive false information such that the attacker can even control the drone. For example, the GCS could provide updated navigation information or even controls if being remotely operated. GPS signals aid the UAS in navigation. UAS swarms communicate to coordinate and avoid collision.

Table 4. Network/Communication Attack Descriptions.

Attack Method	**Description**
Wormhole, Relay Attack	Two malicious nodes can tunnel messages to each other such that messages in the overall system do not take broadcasted or expected routes (i.e. changes the apparent topology of network)
Sybil	An adversary registers fake identities in an ad-hoc network to affect voting outcomes in certain routing protocols such as Flying Ad Hoc Network
Sinkhole	Malicious node may advertise itself as the best route in the network and may modify, drop, or delay packets
De-authentication	Posing as a legitimate entity but sending messages to de-authenticate can cut the link to legitimate entities, for example between UAS and GCS
Packet Sniffing / Eavesdropping	Listening to network communication to gain information about message protocols and patterns can aid in launching network attacks.
Password Breaking	Brute forcing passwords or exploiting protocols that use broken, improperly implemented, or weak authentication methods.
Person in the Middle	Similar to Sinkhole attacks but often the victims believe they are directly

	communicating to each other.
Masquerading	Malicious node attempting to appear as a valid node to gain information or launch additional attacks such as sinkhole, wormhole, etc.
Replay Attack	Captures messages and replays at a different time to launch spoofing or jamming attacks, particularly for when messages are encrypted.
Fuzzing	Fuzzing the protocol to cause software or control algorithm bugs to trigger or reveal information about the message format.
General Jamming	Similar to UAS hardware attacks but more protocol specific. Flooding host's network interface with protocol messages, includes ping floods, Transmission Control Protocol (TCP) handshake flooding, etc. to result in denial-of-service in network

3.5 Server/Cloud Attacks

The information stored in the remote server or cloud can be of interest to attackers. It may include data collected during flight such as flight logs, drone models, video footage, and private information about operators. All this information can aid in launching different types of attacks from the other categories. This category acts more like a traditional software system. Attacking servers connected to the internet is a classic subject of cyber-attacks and existing attacks and countermeasures will also apply to servers for UAS with few differences, if any. For this reason this report will not go into detail for attacks in this category.

3.6 Conclusion

From this survey of attacks on UAS and cyber-physical systems, one can see that the Server/Cloud and GCS components are similar to traditional software systems but the Network/Communication, UAS Hardware, and UAS Software have significantly more unique challenges.

NIST v2 has the core functions of Govern, Identify, Protect, Detect, Respond, and Recover. To properly verify how well UAS is covered by this, the attacks need to be mapped to a viable and practical set of defenses. For example, Govern includes supply chain risk management, which if properly implemented will mitigate threats from supply chain attacks. Protect includes a broad variety of subcategories such as data security, awareness and training, and platform security which

might cover social engineering, exfiltration of sensitive data, and security against software attacks. Detect, Respond, and Recover may cover defenses such as runtime or anomaly detection of spoofing attacks so that a hardened safety mode can be engaged if needed. It is difficult to completely defend against all possible outcomes and attacks, so ideally there needs to be guidance on threat severity and threat intelligence, and an appropriate way to assess risk appetite and consequently identify the necessary defenses.

4 MALWARE SURVEY – SUPPORTING DOCUMENT (DU)

Over the past few decades, software has evolved from being an obscure tool used by few, to a ubiquitous tool used by virtually everyone. While software has had a net positive impact on society, a small subset of users uses it to impact society negatively. The software they write, called "malware," is costly and difficult to detect and mitigate. The malware infects any host they manage to infect, including IoT devices.

The Internet of Things can be described as a network consisting of "smart objects," which are everyday items with Internet connectivity embedded into them to give them remote data sharing capabilities [1]. The number of active IoT devices has risen sharply during the past decade, and as a result, their security is very important. Many devices perform essential tasks that need to be running continuously and uninterrupted, such as security cameras, home locks, heart monitoring devices, and even Unmanned Aerial Vehicles (UAVs). Such devices are the potential targets of malware, as are the components that help power them: gateway devices and the cloud.

There are several common IoT attack models, such as Denial-of-Service or Distributed Denial-of-Service (DoS/DDoS) attacks, jamming, and spoofing [26]. DoS or DDoS attacks refer to when attackers flood the target server (with which the IoT device communicates) with bogus requests, leaving the server unable to fulfill the requests of the IoT device. Jamming refers to when attackers send fake signals to interrupt ongoing communication between the device and the server(s) with whom the device is communicating. This results in a depletion of the device's resources, such as power and/or bandwidth. Lastly, spoofing refers to when attackers impersonate a genuine IoT device to gain unauthorized access to an IoT system in hopes of launching another attack once inside, perhaps a DDoS attack.

Detecting malware attacks can be difficult. For instance, an attacker could embed malware into trusted applications and/or could send malware over protocols that are traditionally allowed by firewalls and access lists [22]. Another problem is that attackers can try to obfuscate their malware or encrypt it, which presents further challenges for someone trying to figure out what is happening on their network [22]. Scale tends to exacerbate these problems. Because of this, an organization with more hosts on the network will generate more network traffic, thus making it even more difficult to manually or automatically scrutinize the large amounts of data [22].

New methods for obfuscating malware have emerged, built on previous methods to make their detection more difficult. One of the first methods used to circumvent traditional anti-virus software was encrypted malware. Encrypted malware makes detection more difficult because they make the bit sequence of the malware binary different than all the bit sequences in the malware signature databases created by anti-virus companies. Another way that adversaries can make malware less detectable is by creating polymorphic malware, which alter the decryption code each time a copy of the malware is created. This succeeds in making the detection process more difficult, but not impossible, because once the code is decrypted, the malware can be analyzed in the computer's local memory. Once adversaries found that polymorphic malware was not the best solution, a new idea emerged: *metamorphic malware*. Metamorphic malware modifies all their malware code, rather than only the decryption code, every time the code is copied during the malware propagation process. Metamorphic malware is less prevalent because they are harder to create but are more alarming due to their ability to bypass anti-virus software.

4.1 The IoT Ecosystem

The IoT ecosystem is divided into four main components: the app, the router or gateway device, the cloud, and the IoT device. Each of these components are important and need to be functioning correctly for the IoT device to work properly and as expected. Figure 2 is an illustration of these components, and each has a section in this document devoted to their roles in the ecosystem.

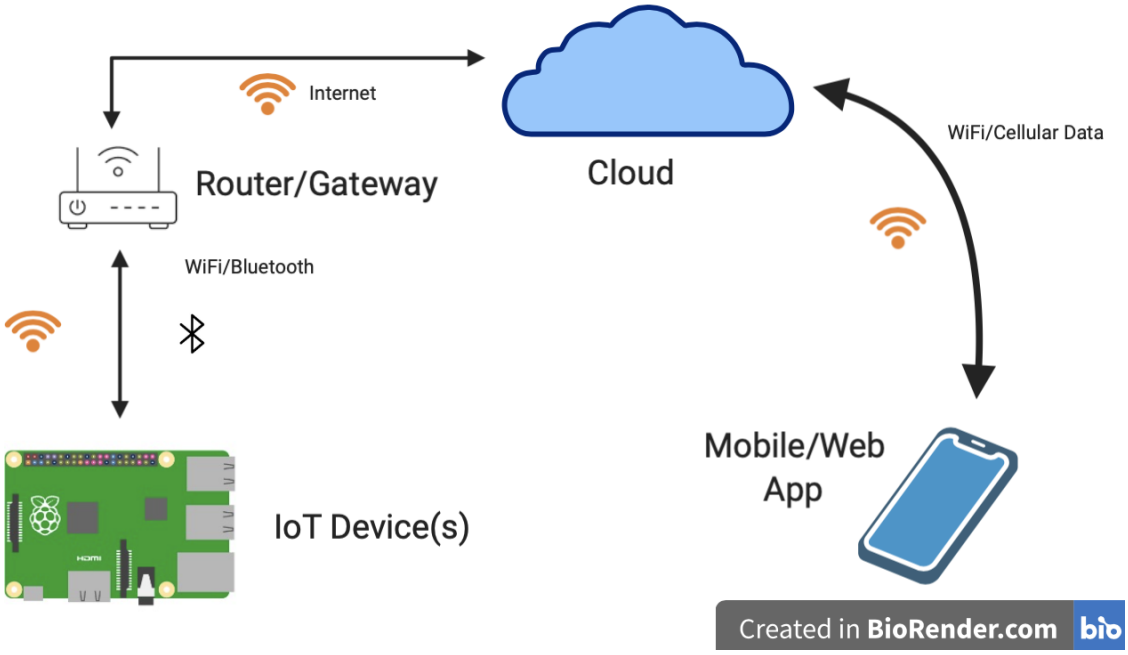

Figure 2. The IoT ecosystem consists of a mobile/web app, an IoT device, a gateway, and the cloud.

4.1.1 App

The app is the part of the IoT ecosystem with which a consumer interacts. Usually either through a web app or a mobile app, these apps are where users can configure their device, change the settings of the IoT device (such as the temperature on a Smart thermometer), and get the information the IoT device is there to collect (*i.e.,* "what is the current temperature of my home?").

The most likely and possible attack vector for malware to manifest itself in āpps is through permissions granted to an app by an operating system. For example, the Android mobile Operating System (OS) has malware due to malicious apps that exploit excessive permissions for certain apps that are available for download [13]. Each app running on the Android OS must declare the permissions it requires to run, which provide access to device functions such as "INTERNET" or "SMS RECEIVED" [13]. Attackers can create malicious apps that declare the permissions they need to run, and thus are granted unneeded access to data such as text messages received or internet activity. These apps could potentially be used to control IoT devices, such as an app that provides the live camera feed of an IP camera.

However, in terms of the IoT ecosystem, this component is less likely than the others to be hacked or targeted for several reasons. The first reason is that the app is likely on a device that the consumer uses regularly, and thus monitors frequently. If something suspicious or malicious is happening on the app, the user is much more likely to spot it rather than something suspicious happening on a device that is likely not near them. The second reason is related to the first. Since the app is probably either on a user's laptop or mobile device, it's more likely to be patched and kept up to date. If the app is running on a phone or tablet, this is probably even more likely, as far fewer malwares can infect a cell phone in contrast with an IoT device running an outdated version of Linux. The user probably will have changed the default passwords and credentials on these devices as well, which is not commonly done on IoT devices. While the app is an important component in any IoT network, for these reasons, this survey focuses more on the other three components of the IoT ecosystem.

4.1.2 Router/Gateway Device

The router is an essential part of the IoT ecosystem, as it allows for the IoT device and the user to be connected. The information exchanged between the device and the user is at the mercy of the information the router allows to be exchanged. As a result, there is an increasing amount of malware targeting the router. One example is when infected routers are recruited to be part of DDoS attacks, like IoT devices being recruited for the same purpose [4].

A router-specific example is malware that forces the router to drop certain packets, making communication difficult or impossible. Often this is accomplished by requesting a packet resubmission when the packet has already been submitted successfully. This action can harm

the IoT device's battery life and diminish the network's throughput and increase its delay time [24]. One way to combat this issue is to secure the gateway device or Access Point (AP), which will then ensure that the communication flowing through it is unhampered. To accomplish this, for example, one may implement an Intrusion Detection System (IDS) on the access point, and let the IDS decide whether the access point is infected or not, as described in [24]. The IDS first keeps track of the number of packets flowing through the IoT device, which includes the packets sent as a result of a NACK (no-acknowledgement) packet sent from the gateway. The access point keeps track of the number of uplink packets successfully received from the gateway. In addition, each IoT device updates the AP regarding the number of packets sent through the non-main channel to the AP at a regular time interval. In prior work, a higher time interval T yields a more accurate classification of the anomaly with a higher probability [24]. This method proved useful for determining if there is an adversary corrupting communication between an IoT device and an access point by way of an infected gateway device.

Securing the router connecting the IoT ecosystem is imperative, since without the router the ecosystem is useless due to none of the devices being able to communicate.

4.1.3 The Cloud

The cloud usually consists of storage on servers belonging to a third party, such as Amazon Web Services, where data is stored. For example, perhaps a user has a Raspberry Pi with a web camera attached acting as a security camera. The Raspberry Pi can transmit the feed to the user, but also to the cloud to save a record of the video data. Although the cloud has malware concerns of its own, usually these issues are monitored by their proprietors, such as Amazon, and are out of the scope of this document. The important aspect of the cloud that pertains to this research is the data retention policies employed by the cloud. In other words, users want to know (and have control over) what data is stored in the cloud, and for how long. The data retention policy will answer these questions and outline the data to keep or delete based on the amount of time it has been available in the cloud [12]. With this comes the problem of proof-of-deletion, which is basically the guarantee to the user that the cloud no longer has access to the data and that is has been permanently and irrecoverably deleted.

4.1.3.1 Docker Hub

A related issue is the security of reusable Docker Hub images. Docker containers have become popular alternatives to traditional virtual machines over the past few years to use applications shared over physical hosts [19]. Because of this, a registry called Docker Hub was created, which acts as a type of cloud application where users can upload and download Docker images. This registry shares both official and community images to users. Official images are public and certified by vendors, such as Oracle or Red Hat, while community images can be created by any user. The sharing of images between users presents a potential security breach in which a user could inject malware into an image that is then shared with

other Docker users without their knowledge of the pre-installed malware. In addition, new images (called child images) can be created from current images (called parent images), which means malware can be embedded in parent images and passed along to numerous child images.

Another reason to be alarmed about possible vulnerabilities with Docker is that it, by default, runs with root privileges [25]. More than 350,000 images were analyzed in current research, and over 180 vulnerabilities were found on average in the images [19]. This research also exposed that the vulnerabilities found in the images often propagated from parent images to child images, similar to how malware are spread in other types of attacks [19].Docker provides a way to certify images by running their inspectDockerImage tool, which minimally checks user-created images for adherence to some basic *best practices* and rules. However, work by Wist *et al.* showed that over 80% of certified images contain at least one critical vulnerability [25]. While there is some mechanism for certifying Docker images, as shown, the current way is not comprehensive. Using machine learning anomaly detection could be a useful avenue of research to explore, as more needs to be done to guarantee the security of images downloaded from Docker Hub.

4.1.4 Device

The IoT device itself is a very important aspect of the ecosystem and is often the target of malware. These devices take many forms and can be anything from a wind meter to a refrigerator to a driving assistant in a car or a UAV.

4.1.4.1 Raspberry Pi

One device that is especially useful in IoT malware detection research is a Raspberry Pi. Raspberry Pi's are small, single-board computers that run a Linux distribution, often the Debian-based Raspbian, as well as other Linux distributions such as Ubuntu. The Raspberry Pi is desirable as a testbed for IoT research primarily for its ease of use and its use of the Linux kernel, as well as its ability to act as many different IoT devices, limited only by the users' configuration. For instance, a Raspberry Pi could be connected to a webcam and become an IP camera that is able to communicate with other hosts via ssh, or it could run downloadable Amazon Alexa software and become a customized AlexaPi [2]. Likewise, Raspberry Pi's can also be used for photography, surveillance, and other tasks when connected to a UAV [17]. As such, many different IoT ecosystems can be created simply by changing the configuration of this one device.

4.1.4.2 IP Camera

A common type of IoT device that is the target of malware is an IP camera, which can be used for tasks such as security or surveillance. In these areas, their security is essential, as well as a guarantee of data integrity. If, for example, an IP camera in a bank is compromised by a looping attack, the camera could capture an actual video feed, and play back this old video recording when the bank is being robbed. Furthermore, any IoT device is susceptible to malware, and while some may be deemed more important than others, any device can be

recruited to take part in a DDoS attack, or other types of coordinated attack.

4.1.4.3 Unpiloted Aerial Vehicle

Another increasingly common IoT device is an UAV. Originally used in military operations, UAVs have become popular for commercial and personal tasks as well due to the decreasing costs to own and operate them as well as their recent technological improvements [10]. They are often used for tasks in agriculture, commercial delivery, media applications, border control, search and rescue, *et cetera*. [15] [16] [17]. Since UAVs have grown in popularity, the interest in attacking them has grown proportionally. The attacks are often focused on the GPS systems guiding the UAVs as well as the data and communications streams between the UAV and the user [7]. Attacks on the GPS systems can include spoofing and jamming attacks, while the possible threat vectors can include errors in configuring communication, sensor, and system settings [7] [16]. It has also been shown that some UAVs are susceptible to man-in-the-middle attacks because of weak Internet security and other vulnerabilities [15]. Lastly, like many IoT devices, UAVs can also fall victim to DoS attacks [16]. A UAV is simply a specialized IoT device, so many of the attacks lodged against a typical IoT device are similarly used against UAVs as well. Since UAVs often perform critical tasks, the security of these devices is extremely important. As their popularity and use continues to grow, so will their vulnerability.

IoT devices can take on many forms, and attacks on these devices can likewise vary. Whether IoT devices are attacked using DDoS or physical attacks, these devices should be set up to withstand a variety of attacks from adversaries. The variety of known attacks will be explained more in subsequent sections of this document.

4.2 Types of Attacks

Attacks on IoT devices are diverse, but usually fall into two broad categories: physical and virtual. Examples of physical attacks include overheating, which involves placing a heat source in close proximity to the victim device in order to overheat it, as well as cutting off power to the IoT device. Virtual attacks are attacks emanating from another computing device and include attacks such as malware. Research by Shi *et al.* identified six different types of attacks on IoT devices [18], and the list of six is far from comprehensive:

1. Viruses - any malware that spreads between hosts by replicating themselves.
2. DoS attacks - attacker overloads component(s) of IoT device, such as the CPU or memory access, leaving it unable to process requests. Trojans - attacker places malicious code into a benign application to gain control of the IoT device in order to, for example, exfiltrate data.
3. Intrusion - attacker tries to gain control of a shell on the victim via ssh. An example of this is a Remote Access Trojan.

4. Power cut - attacker removes the power source from IoT device.

5. Overheating - attacker places a heat source near the victim, causing it to overheat and malfunction.

The main idea presented in a paper by Shi *et al.* to detect this diverse group of attacks is to use energy consumption as a metric to determine whether or not a device is infected [18]. This is to overcome the problem of not being able to trust a (potentially) infected device after it has been compromised by an adversary. It also provides a way to detect both physical and virtual attacks.

Perhaps the most common attack on IoT devices is a DoS/DDoS attack. In a DDoS attack, malware takes over a device and is recruited to be part of a botnet and connects to other malicious IoT devices [22]. A botnet can be described as a group of connected computers recruited to take part in a coordinated task [22]. Once infected, the IoT device may behave normally for a time, but will eventually be used for a malicious purpose: disabling a targeted website or service, for example. One essential part of a DDoS attack is IP spoofing, which is the act of forging the sender's address in the IP header [8]. Specifically, spoofing is used in Volumetric and Reflector DDoS attacks. Volumetric attacks send a large volume of packets to a target. Reflector attacks involve spoofing the IP address of the victim in service requests sent to other servers [8]. The servers then respond to the victim device instead of the desired destination and flood the IoT device. After the victim is flooded with packet data, it may not be able to respond to legitimate requests due to insufficient bandwidth.

4.3 Malware Data Acquisition

One of the main ways to collect data for experimentation in IoT malware detection is to create a honeypot. This acts to lure would-be hackers in order to get their malware code and study it. Often this is accomplished by exploiting lax security on a device, such as using default passwords and ports left open unintentionally. Once the device is attacked, the owners of the honeypot are able to study and replicate the code, thus learning more about the malware targeting their devices. As a result, malware detection and mitigation software can be developed through reverse-engineering the captured malware sample. Since it is now known how the malware infects the device, all that needs to be done is preventing that method from working again. Unfortunately, the problem with this approach is that the creators of the malware will continue to find new ways to infect devices. However, there are instances where one of the families of IoT malware is found, such as Mirai, and thus gives us insight into other kinds of malware due to the similarities between different malware variants.

A honeypot specifically designed for IoT-related malware is an IOT Honeypot (IoTPOT), launched in 2015, which emulates Telnet services of various IoT devices to attract new viruses that use Telnet [11]. According to their research, the most commonly attacked IoT devices are DVRs, IP cameras, and routers. The IoTPOT architecture has a few components, the most important of which is the Frontend Responder, which is responsible for

emulating different IoT devices by handling incoming TCP connection requests, banner instructions, authentication, and command interactions. It then sends these commands to the IoT Sandbox (IoTBOX) backend, which is a set of sandbox environments running different Linux configurations. IoTBOX determines the response to the command request, and forwards it back to the Frontend Responder, which then forwards it to the client. The Profiler, a second component, parses commands between the Frontend Responder and IoTBOX and saves them for later use to reduce the need to communicate with IoTBOX (also subjecting it to fewer malware). The third component is the Downloader, which examines the interactions for download triggers of remote files, such as malware binaries or files obtained from running wget, ftp, *et cetera.* The fourth component is the Manager, which handles configuration of IoTPOT, such as connecting IP addresses with device profiles. During 39 days of data gathering, over 70,000 hosts visited the honeypot [11]. There were three typical stages of attacks:

1. Intrusion - login attempts, in which adversaries try to log into the honeypot to gain access to the device.

2. Infection - discover and change the environment to enable downloading malware. Usually, these activities are automated.

3. Monetization - a command and control server is used to control the device and perform malicious activities, such as a DoS attack or bitcoin mining. The attacker is now able to use the newly recruited device for malicious activity.

Many of the attacks observed when IoTPOT was running were coordinated, in that one compromised host would infiltrate the victim and find out its login credentials and CPU architecture, and then send that information to other hosts so they can attack the victim as well [11]. Most of the attacks observed were UDP floods and different types of TCP floods, which is a type of Denial-of-Service attack in which the attacker overwhelms the target's ports with IP packets containing large datagrams. DNS and SSL attacks were also observed [11].

Another similar project, proposed in the CODASPY 2019 proceedings, increased the chances of their honeypot being attacked by creating multiple virtual private network tunnels forwarding to an IoT device [23]. The usage of a real IoT device lends credibility to the honeypot, and by leaving it completely exposed to hackers, increases the chances of it being attacked. The key to this type of honeypot is to restrict all outside information to the honeypots, such as surrounding Wi-Fi networks, and to set up a firewall to prevent the malware from propagating further on the network [23] [11].

In the last five years, a large amount of data has been collected from these various research projects, especially the IoTPOT project. Although it was conducted in 2015, that project continues to inspire others in the IoT security field and provides a guide for collecting data.

While this data can be used to build more robust malware detection systems, there are still areas in which this data is not comprehensive. For instance, collecting more data in securing the routers and gateway devices that connect the IoT devices, and not just in the securing of the IoT devices themselves, is a useful research avenue to explore. Most of the current data available focuses on securing the IoT devices themselves, without much thought given to securing the routers that connect them to the Internet.

4.4 IoT Malware Detection Methods

Malware detection can generally be approached in two ways: with and without the use of machine learning. Non-machine learning malware detection uses signature analysis. Static analysis often reviews the language and syntax structure while dynamic analysis uses tools such as honeypots to capture malware and study its behaviors [9]. Historically, most malware detection has been signature-based. This method works well on personal computers, but does not work as well on IoT devices, for a variety of reasons. Perhaps the most important reason is that IoT devices constantly contend with a scarcity of resources, as well as a lack of protection against metamorphic malware [3]. This includes memory as well as computing and electrical power. Because of these reasons, machine learning and deep learning methods have become useful due to their high detection rate and low resource consumption. Deep Learning uses a lot of resources to train the model but uses relatively few resources to detect malware after the model has been trained. Traditional machine learning techniques include Support Vector Machines, Logistic Regression, *et cetera,* while deep learning models are usually Artificial Neural Networks (ANNs) or their specializations such as deep ANNs, which include Convolutional Neural Networks. Recently, using Convolutional Neural Networks for anomaly detection has become more common. This process uses gray-scale images of binary files for malware classification. This topic is described in Section 5.2.

4.4.1 Anomaly Detection using Machine Learning

Recently, anomaly detection has become the preferred method for detecting malware on IoT devices due to its limited resource consumption and flexibility. It also has proven successful because of the limited and predictable behavior of IoT devices. IoT devices are usually set up to complete a few specific tasks, and because of this, they often communicate with a limited number of external servers, and their resultant network traffic behavior and execution behavior (via, for example, system calls) is predictable [5]. Anomaly detection also works well for zero-day malware attacks, since anomalies in kernel and network behavior can be detected almost instantaneously. The general anomaly detection pipeline consists of four main steps when collecting captured network traffic data:

1. Traffic capture - record metadata such as the timestamp, protocol, source IP and port, destination IP and port, packet size, and contents. `tcpdump` is useful for recording this data and saving it in a `pcap` file.

2. Group packets by device and time - separated by source IP, then divided into non-overlapping time windows.

3. Feature extraction - determine the most useful metadata to explain the data, such as the destination IP.

4. Binary classification - using ML methods such as ANNs, SVMs, KNN, random forests, decision trees, *et cetera*, to classify data points as benign or malicious.

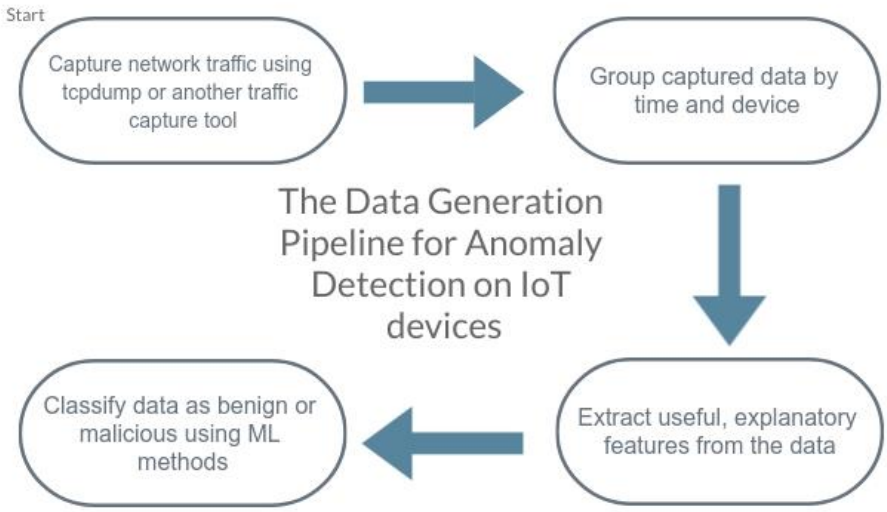

Figure 3. The Data Generation Process used for Anomaly Detection with Network Traffic data.

This type of malware detection works well with the standard IoT ecosystem described above and has been used by multiple research projects. A common ecosystem consists of a Raspberry Pi acting as a router, an IP camera (also possibly implemented as a Raspberry Pi), and any other IoT devices connected to the router, such as a thermostat or a light. There is some feature engineering that can be done to the collected data, and the features fall into two categories: stateless features and stateful features. Stateless features include packet protocol, size, and inter-packet interval, while stateful features include IP destination address cardinality and novelty, and bandwidth [5]. It has been shown that stateless features outperform stateful features in this type of anomaly detection [5].

The features needed for anomaly detection could also be drawn from system data, consisting of a log of system calls made during the data capturing timeframe. On Linux-based IoT devices, the command `ftrace` can be used to record system call information and create the log file [1]. Capturing the system calls during a period of known benign activity, as well as during a time of known malware execution, could provide insight into any connections between malware

running inconspicuously and the system calls executed by the malware. The feature engineering process outlined above would be very similar in this case. In previous work by [1] and [2], a *bag-of-n-grams* approach was used, in which short sequences of system calls during a small period of time are considered. This approach often yields patterns between the system call n-gram sequences that make malware detection easier. In addition, there has also been work done where a combination of both system calls, and network traffic data was captured and used together for feature engineering successfully [2]. In fact, it was shown that a malware detector based on combined system call and network traffic data detected malware better than the system call malware detector or network traffic malware detector did individually [2]. These methods can use Recurrent Neural Networks (RNNs) and LSTMs as well because they work on sequential data n-grams.

4.4.2 Image Recognition for Malware Detection

An alternative IoT malware detection method has emerged recently: using a Convolutional Neural Network (CNN) to classify binaries transformed into gray-scale images. Classifying code binaries in the form of images has proven to be successful, at least in a limited data scope. In work conducted by Su *et al.*, malware samples collected from two malware families, Mirai and Linux.Gafgyt, were able to be classified correctly 94% of the time, with a 5% false positive rate [21]. This research used malware samples collected by the IoTPOT honeypot and were transformed from binaries to gray-scale images by reformatting them into an 8-bit string sequences [21]. A decimal encoding represents the value of a one-channel pixel, which is then formatted into a 64x64 image to be fed into a CNN. Their results indicated that malware images tend to be more dense than benign images [21].

In related work, application binaries are converted into gray-scale images, which are then transformed into sequences of patterns and fed into a RNN [20]. The steps to convert the binary are:

1. Perform raster scanning to find patterns in the image.
2. Use Cosine similarity to distinguish between patterns. The Cosine similarity measures the similarity between two non-zero vectors and is defined to be the Cosine of the angle between them.
3. Convert the image into a sequence of patterns, and feed the result into a RNN.

This approach yielded the same 94% accuracy rate, but a downside of this approach, as discussed by the authors, is the latency that is involved in the image-based malware detection [20].

Convolutional Neural Networks can be difficult, time-consuming, and resource consuming to train well enough to classify accurately. One solution to this problem is to upload the binaries to a cloud application with more resources that can perform the classification and

send the results back to the device. If CNN were running on a large cloud application, it could be trained faster and provide quicker classification results to the IoT device without putting further constraints on the IoT device's resources.

Using image recognition for IoT malware detection is one of the newest fields of research within IoT security, and there are still many problems to mitigate to make it a viable solution on actual IoT devices. Training a normal CNN on a small IoT device seems impractical for the foreseeable future due to IoT resource constraints. As a result, a better solution is needed, and provides another avenue of research in IoT malware detection.

4.5 IoT Malware Mitigation Methods

After detecting malware running on IoT devices, the next step is to mitigate the impact of the malware infection. The risk of malware propagation is especially high in IoT devices, because whenever one device on a network is compromised, it is much easier to continue and infect more devices connected to the network.

One general mitigation idea is to confine the infected nodes and not let the malware spread. The biggest problem with this method, however, is that it often hampers the throughput of the network, thus degrading its performance [14]. This method also presupposes that the malware was detected correctly, which can be difficult considering that malware often try to hide themselves. If the malware successfully decoys themselves, then the confinement method will not be helpful. Similarly, if the detection algorithm produces a false positive, a node will be confined for no reason, which will also likely be detrimental to the network or could cause a denial-of-service. One way to help resolve this problem would be to set a threshold on the amount of throughput required for the network. Given this, the traffic flowing through the infected node can be regulated, and the overall throughput of the network can be tracked. If the traffic restriction results in a throughput that is lower than the required level, the restrictions can be eased until it returns to being above the required level of throughput again [14].

Another method for mitigating malware is a more centralized idea to mitigate the effects of malware on a larger scale, encompassing more than one network. This method connects to a cloud server that collects large amounts of data related to known IoT vulnerabilities. The idea is to connect an "appliance" directly to the IoT device that maintains vulnerability mitigation policies for known Common Vulnerabilities and Exposures (CVEs) of the specific device it protects [6] by connecting to the cloud server and receiving them. Specifically, the security appliance is responsible for three tasks:

1. Communication - receives packets that are addressed to the vulnerable IoT device, processes, and forwards them to the device at the discretion of the vulnerability mitigation policy.

2. Mitigation - called by the communication module. This module will have a list of

vulnerability mitigation policies to execute.

3. Updater - responsible for receiving updates about newly discovered vulnerability mitigation policies for the IoT device.

The other component of the framework is the cloud-based service. This is responsible for the collection and affiliation of CVEs to specific devices, and for the generation and representation of vulnerability mitigation policies [6]. This framework ensures that IoT devices are up to date with recent security updates or patches and prevents exploitation of CVEs. Adopting the framework removes the responsibility of keeping the device up to date from the user. It is also efficient in that the security appliance only protects the IoT device from known vulnerabilities that apply directly to the type of IoT device to which it is connected.

While this framework appears to be a useful solution in theory, there are some drawbacks to note. For instance, in work by Hadar *et al.*, a Raspberry Pi 3 is used as the security appliance to communicate with the cloud server that connects to the IoT device [6]. If the IoT device itself is a Raspberry Pi, which is common, the cost of operating the device has at least doubled due to now needing two Raspberry Pi's. There is also the cost to creating and maintaining the cloud service to stay up to date with CVEs and be able to communicate with the many types of security appliances. In summary, while the idea of having a cloud server and security appliance for each IoT device does seem to be efficient and increase the security of the IoT devices, it is likely not feasible because of the increased overhead that all consumers would need to contribute.

4.6 Conclusion

Research in securing IoT devices and the networks in which they reside is an important, and interesting, area of research. As the number of active IoT devices continues to grow, the importance of their security grows accordingly. This research is also at the intersection of two interesting and timely areas of research: machine learning and cybersecurity.

The use of anomaly detection, either through traditional machine learning models or through RNNs for sequential data, appears to be a viable means for completing this task. Likewise, using both the network traffic data passing through the router, as well as the system calls being executed by the Linux kernel, appear to be the best combination of data for the model to make accurate classifications.

5 OVERSIGHT CONCERNS (ORSU)

In this section, the researchers assess the cyber threats facing UAS and their potential to disrupt National Airspace System (NAS) operations. Research began by constructing a comprehensive picture of the cyber threat landscape relevant to the NAS. Next, the team analyzed the cyber and

physical impacts of compromised UAS, both on the individual systems and their broader implications for NAS safety. Finally, the team investigated how cyber threats targeting UAS could translate into wider-ranging threats for the NAS.

5.1 NAS Components / Attack Surface

The intricate network of the NAS ensures safe and efficient air travel in US Airspace. To analyze potential threats, one must first understand the system's key components. A joint MITRE-Federal Aviation Administration report [25] identified vital elements, emphasizing their criticality to NAS operations:

1. Air Traffic Control Towers: These facilities provide "safe, orderly and expeditious flow of traffic on and in the vicinity of an airport and also provide for the separation of Instrument Flight Rules (IFR) of aircraft in the terminal areas."
2. Control Centers: Responsible for "providing air traffic service to aircraft operating on IFR flight plans within controlled airspace, and principally during the en route phase of flight".
3. Airport Weather Stations: These stations furnish crucial weather information "such as wind, visibility, weather phenomena, etc." for specific airports.
4. Ground/Satellite-Based Navigation: Comprising both ground-based Navigational Aids (NAVAIDs) like ILS and satellite-based systems like GPS, these tools assist pilots in navigating between points.
5. Satellite Surveillance (ADS-B): Leveraging satellite signals, this technology facilitates surveillance of aircraft position.
6. Landing Systems: Including ground-based NAVAIDs like ILS, these systems support safe aircraft landings.
7. Terminal Radar: Utilizing radar technology to track and display aircraft positions, these facilities also offer safety alerts and traffic advisories.
8. Flight Service Stations: These stations handle essential tasks like "providing pilot briefings, flight plan processing, enroute flight advisories, search and rescue services, and assistance to lost aircraft and aircraft in emergency situations."
9. Airline Dispatchers: Playing a vital role in flight planning, they meticulously consider aircraft performance, loading, winds, weather, airspace restrictions, and airport conditions.

5.2 Threats

5.2.1 Threat Actors

Beyond understanding the NAS, identifying potential attackers is crucial for assessing the threat landscape. The recently revised National Strategy for Aviation Security outlined a range of "originators of threats" to the NAS. These threats are:

1. Terrorists
2. Hostile Nation States
3. Criminals
4. Insider Threat
5. Foreign Intelligence Activities and

6. Spread of Infectious Disease

5.2.2 Threat Types

Some common type of threats originating from UAVs include:

1. Disruption of Aviation Activity: Technical malfunctions, operator errors or malicious activity leading to disruption of safe aviation operations.
2. Weaponization: Mounting explosives, chemical or biological agents, acoustic or optical interference, and deliberately crashing the drone as an attack tactic.
3. Hostile Surveillance: Gathering sensitive information and invasion of privacy due to drones capturing images or videos without consent.

5.2.3 Threat Scenarios

With NAS components, potential threat actors and threat types identified, one can now turn to categorizing attack scenarios involving UAS. A systematic approach to generating threats against the NAS can be achieved by considering combinations of threat actors, threat types, and NAS components. For example:

Targeting of an airport's terminal radar systems by a criminal group, by weaponizing UAS to deliberately crash into the physical infrastructure associated with the radar systems.

To ensure this document aligns with its scope, we need to further refine how UAS are used to attack NAS components. Only scenarios where attackers exploit vulnerabilities in the UAS itself through cyber means will be considered. This excludes physically weaponizing UAS. Analyzing the physical and cyber consequences of such cyber-attacks on NAS components is crucial for achieving this refinement.

This section explores potential ways malicious actors could use cyber-attacks to exploit UAS and harm the NAS.

5.2.3.1 Disruption of Aviation Activity:

Scenario 1: Physical Disruption:

- Adversary hijacks a UAS and directs it to a sensitive location: This could include airports, military bases, or other secured facilities.
- The mere presence of the UAS can disrupt operations at these locations.

Scenario 2: Cyber Disruption:

- Adversary hijacks a UAS and equips it with malicious tools such as hardware WiFi jammer or advanced cyber exploitation toolkits.
- This can interrupt critical communication and compromise hosts in the network being targeted.

5.2.3.2 Weaponization:

Scenario 1: Mid-air collision with manned aircraft:

- A malicious actor launches a UAV targeting a manned aircraft to interfere with safe flight operations.

Scenario 2: Denial-of-service attacks:

- Adversary launches cyberattacks to disable a legitimate drone's control system. This can result in losing control and crashing into people, locations, NAS infrastructure, and other critical infrastructure.

5.2.3.3 Hostile Surveillance

Scenario 1: Hijacked UAS for unauthorized surveillance:

- A malicious actor intercepts a benign UAS mid-mission and redirects it to an unauthorized area for covert data collection.
- The gathered information is transmitted to the attacker's device or uploaded to a remote server.

Scenario 2: Exploiting authorized surveillance:

- A UAS performing authorized surveillance in a secure location is compromised, allowing the attacker to access and capture the collected footage.
- This footage is then relayed to the attacker's control device or uploaded to a remote server.

5.3 Takeaway

In summary, by mapping cyber threats to both UAS vulnerabilities and NAS, one can generate realistic scenarios showcasing the potential dangers of integrating UAS into the NAS landscape to help determine oversight requirements.

Table 5. Attack Threat Classification.

Attack		UAS Phases of Operation																				Threat Types		
		Pre-Flight / Mission Planning		Preparation /System Checks (applicable at almost all phases of mission/flight)						Launch					Mission/Application/Flight (Communication)			Return to Land		Post- Flight		Others			
Attack Reference Number	Attack Name	Flight Planning (both for manual and autonomous)	Programming flight (autonomous only)	Ground station	Flight controls	Data links	GPS	Sensor	Power - battery/fuel	System checks (similar to those noted above)	Altimeter verification	Flight	Manual	Autonomous - Flight plan verification	Data Relay - Telemetry	Payload data - Video relay	Payload data - Sensor Information	Manual	Autonomous	Ground Station	Data Download	Emergency Procedures	Disruption of Aviation Activity	Weponization	Hostile Surveillance
NLx	**Network Attack**																								
NL1	Black Hole/Gray Hole	Low	Low	Low	Low	Low	Low	Low	Low	Low	Medium	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
NL2	Wormhole	Low	Low	Low	Low	Low	Low	Low	Low	Low	Medium	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
NL3	Sybil	Low	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Medium	Medium	Medium	Low	Low	Low	Medium	Medium	Low	Low	Medium	✓	✓	-
NL4	Sinkhole	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Medium	*High/M	*High/M	*High/M	Low	Low	Low	*High/M	*High/M	Low	Low	*High/M	✓	✓	-
NL5	Radio Frequency (RF)-based	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
NL6	Protocol-based Jamming	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
NL7	Deauthentication	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Medium	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
NL8	Packet Sniffing/Analysis	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	-	-	✓
NL9	Password Breaking	Low	Low	High	Low	Low	Low	Low	Low	High	Low	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	✓
NL10	Person-In-The-Middle	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	High	High	Low	Low	Low	✓	✓	✓
NL11	Command Injection	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Low	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	✓
NL12	Masquerading	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	Low	Low	High	✓	✓	✓
NL13	Replay Attack	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	*High/M	*High/M	Low	Low	High	✓	✓	✓
NL14	Relay Attack	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	*High/M	*High/M	*High/M	Low	Low	Low	*High/M	*High/M	Low	Low	*High/M	-	-	✓
NL15	Fuzzing	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	✓
SWx	**Software**																								
SW1	Code Injection	Low	Low	Low	Low	Low	Low	Low	Low	High	*High/M	*High/M	*High/M	*High/M	Low	Low	Low	*High/M	*High/M	Low	Low	*High/M	✓	✓	✓
SW2	Database Injection	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	✓
SW3	Firmware Modification	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Medium	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
SW4	Sleep Deprivation	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	✓	✓	-
SW5	Buffer Overflow	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	High	High	Low	Low	Medium	✓	✓	-
SW6	Malware infection	Medium	Medium	Medium	Medium	Medium	Medium	Medium	High	High	High	High	High	High	Medium	Medium	Medium	High	High	High	Medium	High	✓	✓	-
SW7	Supply Chain Attack	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	✓
HWx	**Hardware**																								
HW1	Spoofing - GPS	Low	Low	Low	Low	Low	Low	Low	Low	Medium	High	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
HW2	Spoofing - Other Sensors	Low	Low	Low	Low	Low	Low	Low	Low	Medium	High	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
HW3	Spoofing - ADS-B, Remote ID	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Medium	Medium	Medium	Medium	Low	Low	Low	Medium	Medium	Medium	Medium	Medium	✓	✓	-
HW4	Spoofing - Actuator	Low	Low	Low	Low	Low	Low	Low	Low	Medium	High	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
HW5	Jamming - GPS	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
HW6	Jamming - Other Sensors	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
HW7	Jamming - ADS-B, Remote ID	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Medium	Medium	Medium	Medium	Low	Low	Low	Medium	Medium	Medium	Medium	Medium	✓	✓	-
HW8	Jamming - Actuator	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	-
HW9	Firmware Flashing	Medium	Medium	Medium	Medium	Medium	Medium	High	High	Medium	Medium	Medium	Medium	Medium	Medium	Medium	Medium	High	High	Medium	Medium	High	✓	✓	✓
HW10	Supply Chain Attack	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	✓	✓
GCSx	**GCS**																								
GCS1	Remote access	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Medium	Medium	Low	Low	Low	High	High	Low	Low	High	✓	✓	✓
GCS2	Forced quitting application	Low	Low	Medium	High	High	Medium	High	Medium	High	High	High	High	High	Low	Low	High	High	High	Low	Low	High	✓	✓	✓
GCS3	Data exfiltration	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	✓	-	✓
GCS4	Password Breaking	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Medium	Medium	Low	Low	Low	Medium	Medium	Low	Low	High	✓	-	✓
GCS5	Reverse Engineering GCS	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	High	High	High	Low	Low	Low	High	High	Low	Low	High	✓	-	✓
GCS6	Social Engineering	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Medium	High	High	Low	Low	Low	Medium	Medium	Low	Low	High	✓	-	✓
SRVx	**Server**																								
SRV1	Data leakage	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	-	-	✓
SRV2	Pilot identity leakage	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	Low	-	-	✓
SRV3	Location leakage	Low	Low	Low	Low	Low	Low	Low	Low	Medium	Low	High	Medium	High	Low	Low	Low	High	High	Low	Low	Medium	-	-	✓

6 CONCLUSION

There is no UAS specific framework for identifying and mitigating cybersecurity concerns. Furthermore, while general purpose frameworks are necessary for UAS deployment they are not sufficient due to lack of focus on cyberphysical issues. While traditional computing systems may have sensors and actuators that interface with the physical world, they are dominant in UAS where they define a significant attack surface.

The researchers will address the need for a UAS framework by focusing on cyberphysical issues common among UAS with a goal of protecting the airspace. Rather than reinvent a framework for traditional cybersecurity issues, the researchers will focus on cyberphysical issues that characterize UAS. Starting with the A38 literature study, the team will identify the most critical and likely attacks. The team will then develop a framework that addresses those attacks focusing on cyberphysical security issues.

7 REFERENCES

[1] N. An, A. Duff, G. Naik, M. Faloutsos, S. Weber, and S. Mancoridis. 2017. Behavioral anomaly detection of malware on home routers. In *2017 12th Intenational Conference on Malicious and Unwanted Software (MALWARE)*. 47–54. https://doi.org/10.1109/MALWARE.2017.8323956

[2] N. An, A. Duff, M. Noorani, S. Weber, and S. Mancoridis. 2018. Malware Anomaly Detection on Virtual Assistants. In *2018 13th International Conference on Malicious and Unwanted Software (MALWARE)*. IEEE Computer Society, Los Alamitos, CA, USA, 124–131. https://doi.org/10.1109/ MALWARE.2018.8659366

[3] Shiven Chawla and Geethapriya Thamilarasu. 2018. Security as a service: real- time intrusion detection in internet of things. *Proceedings of the Fifth Cybersecurity Symposium* (2018).

[4] Ahmad Darki, Alexander Duff, Zhiyun Qian, Gaurav Naik, Spiros Mancoridis, and Michalis Faloutsos. 2016. "Don't Trust Your Router: Detecting Compromised Routers". In *CoNEXT '16*. Irvine, CA, USA.

[5] Rohan Doshi, Noah Apthorpe, and Nick Feamster. 2018. Machine Learning DDoS Detection for Consumer Internet of Things Devices. 2018 IEEE Security *and Privacy Workshops (SPW)* (May 2018). https://doi.org/10.1109/spw. 2018.00013

[6] Noy Hadar, Shachar Siboni, and Yuval Elovici. 2017. A Lightweight Vulnerability Mitigation Framework for IoT Devices. In *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy* (Dallas, Texas, USA) *(IoTSamp;P '17)*. Association for Computing Machinery, New York, NY, USA, 71–75. https: //doi.org/10.1145/3139937.3139944

[7] C. G. Leela Krishna and Robin R. Murphy. 2017. A review on cybersecurity vulnerabilities for Unpiloted aerial vehicles. In *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. 194–199. https://doi.org/10.1109/SSRR.2017.8088163

[8] Jelena Mirkovic, Erik Kline, and Peter Reiher. 2017. RESECT: Self-Learning Traffic Filters for IP Spoofing Defense. In *Proceedings of the 33rd Annual Computer Security Applications Conference* (Orlando, FL, USA) *(ACSAC 2017)*. Association for Computing Machinery, New York, NY, USA, 474–485. https: //doi.org/10.1145/3134600.3134644

[9] Youness Mourtaji, Mohammed Bouhorma, and Daniyal Alghazzawi. 2019. Intelligent Framework for Malware Detection with Convolutional Neural Net- work. In *Proceedings of the 2nd International Conference on Networking, In- formation Systems amp; Security* (Rabat, Morocco) *(NISS19)*. Association for Computing Machinery, New York, NY, USA, Article 7, 6 pages. https:

//doi.org/10.1145/3320326.3320333

[10] Weina Niu, Jian'An Xiao, Xiyue Zhang, Xiaosong Zhang, Xiaojiang Du, Xiaom- ing Huang, and Mohsen Guizani. 2021. Malware on Internet of UAVs Detection Combining String Matching and Fourier Transformation. *IEEE Internet of Things Journal* 8, 12 (2021), 9905–9919. https://doi.org/10.1109/JIOT.2020.3029970

[11] Yin Minn Pa Pa, Shogo Suzuki, Katsunari Yoshioka, Tsutomu Matsumoto, Takahiro Kasama, and Christian Rossow. 2015. IoTPOT: Analysing the Rise of IoT Compromises. In *Proceedings of the 9th USENIX Conference on Offensive Technologies* (Washington, D.C.) *(WOOT'15)*. USENIX Association, USA, 9.

[12] Nisha Panwar, Shantanu Sharma, Peeyush Gupta, Dhrubajyoti Ghosh, Sharad Mehrotra, and Nalini Venkatasubramanian. 2020. IoT Expunge: Implementing Verifiable Retention of IoT Data. In *Proceedings of the Tenth ACM Confer- ence on Data and Application Security and Privacy* (New Orleans, LA, USA) *(CODASPY '20)*. Association for Computing Machinery, New York, NY, USA, 283–294. https://doi.org/10.1145/3374664.3375737

[13] M. Ping, B. Alsulami, and S. Mancoridis. 2016. On the effectiveness of application characteristics in the automatic classification of malware on smartphones. In *2016 11th International Conference on Malicious and Unwanted Software (MALWARE)*. 1–8. https://doi.org/10.1109/MALWARE.2016.7888732

[14] S. M. Pudukotai Dinakarrao, H. Sayadi, H. M. Makrani, C. Nowzari, S. Rafatirad, and H. Homayoun. 2019. Lightweight Node-level Malware Detection and Network-level Malware Confinement in IoT Networks. In *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*. 776–781.

[15] Nils Miro Rodday, Ricardo de O. Schmidt, and Aiko Pras. 2016. Exploring security vulnerabilities of unmanned aerial vehicles. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*. 993–994. https://doi.org/10.1109/NOMS.2016.7502939

[16] Alessio Rugo, Claudio A. Ardagna, and Nabil El Ioini. 2022. A Security Review in the UAVNet Era: Threats, Countermeasures, and Gap Analysis. *ACM Comput. Surv.* 55, 1, Article 21 (jan 2022), 35 pages. https://doi.org/10.1145/3485272

[17] Arnab Kumar Saha, Jayeeta Saha, Radhika Ray, Sachet Sircar, Subhojit Dutta, Soummyo Priyo Chattopadhyay, and Himadri Nath Saha. 2018. IOT-based drone for improvement of crop quality in agricultural field. In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. 612–615. https://doi.org/10.1109/CCWC.2018.8301662

[18] Yang Shi, Fangyu Li, WenZhan Song, Xiang-Yang Li, and Jin Ye. 2019. Energy Audition Based Cyber-Physical Attack Detection System in IoT. In *Proceedings of the ACM Turing Celebration Conference - China* (Chengdu, China) *(ACM TURC '19)*. Association for Computing Machinery, New York, NY, USA, Article 27, 5 pages. https://doi.org/10.1145/3321408.3321588

[19] Rui Shu, Xiaohui Gu, and William Enck. 2017. A Study of Security Vulnerabili- ties on Docker Hub. In *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy* (Scottsdale, Arizona, USA) *(CODASPY '17)*. Association for Computing Machinery, New York, NY, USA, 269–280. https://doi.org/10.1145/3029806.3029832

[20] S. Shukla, G. Kolhe, P. Sai Manoj, and S. Rafatirad. 2019. Work-in-Progress: MicroArchitectural Events and Image Processing-based Hybrid Approach for Robust Malware Detection. In *2019 International Conference on Compliers, Architectures and Synthesis for Embedded Systems (CASES)*. 1–2.

[21] J. Su, D. V. Vasconcellos, S. Prasad, D. Sgandurra, Y. Feng, and K. Sakurai. 2018. Lightweight Classification of IoT Malware Based on Image Recognition. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 02. 664–669.

[22] Luis Suastegui Jaramillo. 2018. Malware Detection and Mitigation Techniques: Lessons Learned from Mirai DDOS Attack. *Journal of Information Systems Engineering Management* 3(07 2018).https://doi.org/10.20897/jisem/2655

[23] N. An, A. Duff, G. Naik, M. Faloutsos, S. Weber, and S. Mancoridis. 2017. Behavioral anomaly detection of malware on home routers. In *2017 12th Inter- national Conference on Malicious and Unwanted Software (MALWARE)*. 47–54. https://doi.org/10.1109/MALWARE.2017.8323956

[24] N. An, A. Duff, M. Noorani, S. Weber, and S. Mancoridis. 2018. Malware Anomaly Detection on Virtual Assistants. In *2018 13th International Conference on Malicious and Unwanted Software (MALWARE)*. IEEE Computer Society, Los Alamitos, CA, USA, xa;124xa;–xa;131xa;. https://doi.org/10.1109/ MALWARE.2018.8659366

[25] Shiven Chawla and Geethapriya Thamilarasu. 2018. Security as a service: real- time intrusion detection in internet of things. *Proceedings of the Fifth Cyberse- curity Symposium* (2018).

[26] Ahmad Darki, Alexander Duff, Zhiyun Qian, Gaurav Naik, Spiros Mancoridis, and Michalis Faloutsos. 2016. "Don't Trust Your Router: Detecting Compro- mised Routers". In *CoNEXT '16*. Irvine, CA, USA.Rohan Doshi, Noah Apthorpe, and Nick Feamster. 2018. Machine Learning DDoS Detection for Consumer Internet of Things Devices. *2018 IEEE Security and Privacy Workshops (SPW)* (May 2018). https://doi.org/10.1109/spw. 2018.00013

[27] Noy Hadar, Shachar Siboni, and Yuval Elovici. 2017. A Lightweight Vulnerability Mitigation Framework for IoT Devices. In *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy* (Dallas, Texas, USA) *(IoTSamp;P '17)*. Association for Computing Machinery, New York, NY, USA, 71–75. https://doi.org/10.1145/3139937.3139944

[28] C. G. Leela Krishna and Robin R. Murphy. 2017. A review on cybersecurity vulnerabilities for unmanned aerial vehicles. In *2017 IEEE International Sym- posium on Safety, Security and Rescue Robotics (SSRR)*. 194–199. https://doi.org/10.1109/SSRR.2017.8088163

[29] Jelena Mirkovic, Erik Kline, and Peter Reiher. 2017. RESECT: Self-Learning Traffic Filters for IP Spoofing Defense. In *Proceedings of the 33rd Annual Com- puter Security Applications Conference* (Orlando, FL, USA) *(ACSAC 2017)*. Association for Computing Machinery, New York, NY, USA, 474–485. https://doi.org/10.1145/3134600.3134644

[30] Youness Mourtaji, Mohammed Bouhorma, and Daniyal Alghazzawi. 2019. In- telligent Framework for Malware Detection with Convolutional Neural Net- work. In *Proceedings of the 2nd International Conference on Networking, In- formation Systems amp; Security* (Rabat, Morocco) *(NISS19)*. Association for Computing Machinery, New York, NY, USA, Article 7, 6 pages. https://doi.org/10.1145/3320326.3320333

[31] Weina Niu, Jian'An Xiao, Xiyue Zhang, Xiaosong Zhang, Xiaojiang Du, Xiaom- ing Huang, and Mohsen Guizani. 2021. Malware on Internet of UAVs Detec- tion Combining String Matching and Fourier Transformation. *IEEE Internet of Things Journal* 8, 12 (2021), 9905–9919. https://doi.org/10.1109/JIOT.2020.3029970

[32] Yin Minn Pa Pa, Shogo Suzuki, Katsunari Yoshioka, Tsutomu Matsumoto, Takahiro Kasama, and Christian Rossow. 2015. IoTPOT: Analysing the Rise of IoT Compromises. In *Proceedings of the 9th USENIX Conference on Offen- sive Technologies* (Washington, D.C.) *(WOOT'15)*. USENIX Association, USA, 9.

[33] Nisha Panwar, Shantanu Sharma, Peeyush Gupta, Dhrubajyoti Ghosh, Sharad Mehrotra, and Nalini Venkatasubramanian. 2020. IoT Expunge: Implementing Verifiable Retention of IoT Data. In *Proceedings of the Tenth ACM Confer- ence on Data and Application Security and Privacy* (New Orleans, LA, USA) *(CODASPY '20)*. Association for Computing Machinery, New York, NY, USA, 283–294. https://doi.org/10.1145/3374664.3375737

[34] M. Ping, B. Alsulami, and S. Mancoridis. 2016. On the effectiveness of applica- tion characteristics in the automatic classification of malware on smartphones. In *2016 11th International Conference on Malicious and Unwanted Software (MALWARE)*. 1–8. https://doi.org/10.1109/MALWARE.2016.7888732

[35] S. M. Pudukotai Dinakarrao, H. Sayadi, H. M. Makrani, C. Nowzari, S. Rafati- rad, and H. Homayoun. 2019. Lightweight Node-level Malware Detection and Network-

level Malware Confinement in IoT Networks. In *2019 Design, Automa- tion Test in Europe Conference Exhibition (DATE)*. 776–781.

[36] Nils Miro Rodday, Ricardo de O. Schmidt, and Aiko Pras. 2016. Explor- ing security vulnerabilities of unmanned aerial vehicles. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*. 993–994. https://doi.org/10.1109/NOMS.2016.7502939

[37] Alessio Rugo, Claudio A. Ardagna, and Nabil El Ioini. 2022. A Security Review in the UAVNet Era: Threats, Countermeasures, and Gap Analysis. *ACM Com- put. Surv.* 55, 1, Article 21 (jan 2022), 35 pages. https://doi.org/10.1145/3485272

[38] Arnab Kumar Saha, Jayeeta Saha, Radhika Ray, Sachet Sircar, Subhojit Dutta, Soummyo Priyo Chattopadhyay, and Himadri Nath Saha. 2018. IOT-based drone for improvement of crop quality in agricultural field. In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. 612–615. https://doi.org/10.1109/CCWC.2018.8301662

[39] Yang Shi, Fangyu Li, WenZhan Song, Xiang-Yang Li, and Jin Ye. 2019. Energy Audition Based Cyber-Physical Attack Detection System in IoT. In *Proceedings of the ACM Turing Celebration Conference - China* (Chengdu, China) *(ACM TURC '19)*. Association for Computing Machinery, New York, NY, USA, Article 27, 5 pages. https://doi.org/10.1145/3321408.3321588

[40] Rui Shu, Xiaohui Gu, and William Enck. 2017. A Study of Security Vulnerabili- ties on Docker Hub. In *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy* (Scottsdale, Arizona, USA) *(CODASPY '17)*. Association for Computing Machinery, New York, NY, USA, 269–280. https://doi.org/10.1145/3029806.3029832

[41] S. Shukla, G. Kolhe, P. Sai Manoj, and S. Rafatirad. 2019. Work-in-Progress: MicroArchitectural Events and Image Processing-based Hybrid Approach for Robust Malware Detection. In *2019 International Conference on Compliers, Ar- chitectures and Synthesis for Embedded Systems (CASES)*. 1–2.

[42] J. Su, D. V. Vasconcellos, S. Prasad, D. Sgandurra, Y. Feng, and K. Sakurai. 2018. Lightweight Classification of IoT Malware Based on Image Recognition. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 02. 664–669.

[43] Luis Suastegui Jaramillo. 2018. Malware Detection and Mitigation Techniques: Lessons Learned from Mirai DDOS Attack. *Journal of Information Systems Engineering Management* 3 (07 2018).https://doi.org/10.20897/jisem/2655

[44] Amit Tambe, Yan Lin Aung, Ragav Sridharan, Mart´ın Ochoa, Nils Ole Tippen- hauer, Asaf Shabtai, and Yuval Elovici. 2019. Detection of Threats to IoT Devices Using Scalable VPN-Forwarded Honeypots. In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy* (Richardson, Texas, USA)

(CODASPY '19). Association for Computing Machinery, New York, NY,USA, 85–96. https://doi.org/10.1145/3292006.3300024

[45] Nalam Venkata Abhishek, Anshoo Tandon, Teng Joon Lim, and Biplab Sik- dar. 2018. Detecting Forwarding Misbehavior In Clustered IoT Networks. In *Proceedings of the 14th ACM International Symposium on QoS and Security for Wireless and Mobile Networks* (Montreal, QC, Canada) *(Q2SWinet'18)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3267129.3267147

[46] Katrine Wist, Malene Helsem, and Danilo Gligoroski. 2020. Vulnerability Anal- ysis of 2500 Docker Hub Images. *ArXiv* abs/2006.02932 (2020).

[47] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu. 2018. IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security? *IEEE Signal Processing Magazine* 35, 5 (2018), 41–49.

[1]