DOT/FAA/TC-693KA8-19-D-0003/Task 14

Federal Aviation Administration
William J. Hughes Technical Center
Aviation Research Division
Atlantic City International Airport
New Jersey 08405

Part 3 Cybersecurity Data Science Aviation Architecture Framework (CSDS AAF) - System Guidance Document

April 12, 2023

U.S. Department of Transportation
Federal Aviation Administration

NOTICE

This document is disseminated under the sponsorship of the U.S. Department of Transportation in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof. The U.S. Government does not endorse products or manufacturers. Trade or manufacturers' names appear herein solely because they are considered essential to the objective of this report. The findings and conclusions in this report are those of the author(s) and do not necessarily represent the views of the funding agency. This document does not constitute FAA policy. Consult the FAA sponsoring organization listed on the Technical Documentation page as to its use.

This report may be made available upon request to the FAA Aviation Research Division.

Technical Report Documentation Page

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

1. Report No.

DOT/FAA/TC-693KA8-19-D-0003/Task 14 D16d

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle

Title of Report: Cyber Security Data Science- Aviation Architecture Framework

Subtitle of Report: System Guidance Document

5. Report Date

April 2023

6. Performing Organization Code

7. Author(s)

Center for Airspace Resilience (CAR)

Jayson Clifford
Dr. Ilhan Akbas
Lauren Warner
Isidore Venetos
Daniel Diessner

8. Performing Organization Report No.

9. Performing Organization Name and Address

Next-Generation ERAU Applied Research Lab (NEAR)
Embry-Riddle Aeronautical University
1 Aerospace Blvd. Daytona Beach, FL 32114-3900

10. Work Unit No. (TRAIS)

11. Contract or Grant No.

693KA8-19-D-0003/Task 14

12. Sponsoring Agency Name and Address

Federal Aviation Administration
William J. Hughes Technical Center
Aviation Research Division
Atlantic City International Airport
New Jersey 08405

13. Type of Report and Period Covered

14. Sponsoring Agency Code

ANG-E271

15. Supplementary Notes

16. Abstract

This document is the third part of a series of three core documents to provide an overview of the top-down output from the FAA Cybersecurity Data Science Aviation Architecture Framework (CSDS AAF) research program. The intent of this document is to be used as a general guidance reference for the implementation of CSDS AAF in various systems across the aviation ecosystem and to be used as a reference for industry standards or guidance activities. The three core CSDS AAF documents are:

- **Part 1 CSDS AAF - Utilization Strategy**: The primary purpose is to communicate to aviation stakeholders the vision and potential value of the FAA CSDS research and generally how it could potentially be leveraged to address key aviation cybersecurity challenges.
- **Part 2 CSDS AAF - Technical Specification Document**: As an ontology for the CSDS Aviation Architecture Framework, this document provides narrative to describe and explain all of the key AAF components and functions, coupled with diagrams to illustrate the overall AAF structure.
- **Part 3 CSDS AAF - System Guidance Document**: This document provides guidance for implementation of the CSDS AAF which is defined in the AAF Technical Specification Document.

17. Key Words

Security Data Science (CSDS), Aviation Architecture Framework (AAF), Guidelines, Cyber Analytical Cell, Data Life Cycle, Interconnected Individual Systems

18. Distribution Statement

This report may be made available upon request to the FAA Aviation Research Division.

19. Security Classif. (of this report)

Unclassified

20. Security Classif. (of this page)

Unclassified

21. No. of Pages

22. Price

Contents

Figures

Tables

No table of figures entries found.

Acronyms

Acronym	Definition
AAF	Aviation Architectural Framework
AI/ML	Artificial Intelligence/Machine Learning
AWS	Amazon Web Service
CAC	Cyber Analytical Cell
CBOR	Concise Binary Object Representation
CSDS	Cybersecurity Data Science
CJIS	Criminal Justice Information Systems
DAS	Data Acquisition Sensors
DDoS	Distributed Denial-of-Service
DEP	Data Egress Point
DoD	Department of Defense
DTD	Document Type Definition
HSM	Hardware Security Module
IIS	Interconnected Individual Systems
IIoT	Industrial Internet of Things
IaaS	Infrastructure as a Service
ISP	In-System Programming
IT	Information Technology
KBA	Knowledge-Based Authentication
NAS	Network-Attached Storage
OEM	Original Equipment Manufacturer
OT	Operation Technology
OTA	Over-The-Air
OTP	One-Time-Passwords
PaaS	Platform as a Service
QoS	Quality of Service
RAID	Redundant Arrays of Independent Disks
S3	Simple Storage Service
SaaS	Software as a Service
SAN	Storage Area Network
SRG	Security Requirements Guide

TAP	Test Access Port
TSBD	Time-Based Database
TLS	Transport Layer Security
ZTA	Zero-Trust Architecture

Executive summary

A critical challenge in cybersecurity is determining if a cyber incident has or is happening. Data science holds promise to more quickly and more effectively find anomalous data that could indicate a cyber incident. The Cybersecurity Data Science (CSDS) Aviation Architecture Framework (AAF) seeks to apply data science to the Aviation Ecosystem, which involves both Information Technology (IT) and Operation Technology (OT) with various stakeholders such as airlines, airports, and Original Equipment Manufacturers (OEMs). This document defines the conceptual elements and taxonomy of the CSDS AAF. The CSDS AAF Systems Architecture applies this framework in different Environments of Operation and involves Stakeholder Data-Stores, Cyber Analytical Cells (CAC), and Interconnected Individual Systems (IIS). The framework also introduces the CSDS AAF Data Life Cycle, which consists of Acquire, Pre-Analyzed, Collect, Advanced Analytics, and Information Sharing. A critical component in this data perspective is collecting the appropriate data which is described using the Data Sphere concept. The document then describes how to apply and refine the CSDS AAF for a specific Environment of Operation. A chief goal of this document is to help inform future regulatory and standards activities by providing and maturing the CSDS aviation architecture framework.

1 Introduction

This is the third part of a series of three (3) core documents to provide an overview of the top-down output from the FAA Cybersecurity Data Science Aviation Architecture Framework (CSDS AAF) research program. The intent of this document is to be used as a general guidance reference for the implementation of CSDS AAF in various systems across the aviation ecosystem and to be used as a reference for industry standards or guidance activities. The three (3) core CSDS AAF documents are:

- **Part 1 CSDS AAF - Utilization Strategy**: The primary purpose is to communicate to aviation stakeholders the vision and potential value of the FAA CSDS research and generally how it could potentially be leveraged to address key aviation cybersecurity challenges.
- **Part 2 CSDS AAF - Technical Specification Document**: As an ontology for the CSDS Aviation Architecture Framework, this document provides a narrative to describe and explain all of the key AAF components and functions, coupled with diagrams to illustrate the overall AAF structure.
- **Part 3 CSDS AAF - System Guidance Document**: This document provides guidance for the implementation of the CSDS AAF, which is defined in the AAF Technical Specification Document.

2 Cybersecurity Data Science Aviation Architecture Framework (CSDS AAF)

The structure of Parts 1 and 2 of the CSDS AAF documents has been replicated in Part 3. All three (3) parts are intended to provide the complete definition of the CSDS AAF. Some sections in Part 3 may not contain any information if it has already been sufficiently described in Part 2 *CSDS AAF Technical Specification Document* (AAF-2). In those cases, the section headings are left in place to maintain the structure across all three parts of the document, and the previous part of the document will be referenced.

2.1 CSDS AAF Conceptual Elements

The conceptual elements can be found in *AAF-2*.

2.2 CSDS AAF Taxonomy & Reference Model

The taxonomy and reference model definitions can be found in *AAF-2*.

2.3 CSDS AAF Systems Architecture: The Systems Perspective

This section will provide guidance statements for implementing the various systems within the CSDS AAF. The information in these sections is intended to be generic to any environment of operation or use case. Most sections below are organized into three parts, each focusing on a particular aspect of the system: functions, performance, and security. Some sections may not contain guidance on a specific topic if it is not applicable or if it has already been sufficiently described in the *Part 1 CSDS AAF Utilization Strategy (AAF-1)* or *Part 2 CSDS AAF Technical Specification Document (AAF-2).*

2.3.1 AAF Operational Concept (System Architecture)

The AAF operational scenario and concept can be found in AAF-1 and AAF-2 respectively.

2.3.2 System Architecture Aspects of Environments of Operations

2.3.2.1 Interconnected Individual Systems (IIS)

The IIS is a foundational element connected to several architectural components of the CSDS AAF, such as data acquisition sensors, local storage, and data egress points. When answering the three key CSDS objective questions – Is there a cyber-event pending? Is there an attack occurring now? Was an incident/event caused by cyber activity? – the object of the question will often be an IIS. Data acquisition sensors (DAS) can be placed in the IIS to collect, filter, and pre-analyze data. The data is placed in local storage and sent via specialized data egress points to one or more Data-Stores within the environment of operations. This data reflects the current and past state of components within the IIS. The data is used in the CSDS AAF as input to the Cyber-Analytical Cells (CAC).

2.3.2.1.1 Functions

What are the functions of an IIS with respect to the AAF? As defined by the AAF taxonomy, it is the basic building block within any particular environment of operation. It’s a foundational component capable of capturing, storing, filtering, and sending relevant data. It is recommended that the IIS and its components are inventoried and documented. Some considerations when documenting the IIS:

- If already documented, where is this information stored currently?
- How would the information reference the IIS and its components inside analytical models?
- Who is responsible for administering the components? Is there shared ownership of the data in/about the IIS between different stakeholders within the environment of operation?

- Do the data owners have sufficiently granular control and visibility into these systems to inform an analytical model?

This information should be accessible in some form to the CAC, ideally referenced in metadata by a globally unique identifier for each component (i.e., UUID, as defined in ISO 9834-8:2014 or IETF-4122) to allow for traceability. This identifier can be used as a reference in models and data structures linking all associated data to a particular IIS. Some examples of CSDS-relevant component documentation are listed below:

- Serial and model numbers of components, current firmware and software versions, network names, and addresses.
- Component owners and those responsible and accountable for administrating the components.
- The physical connections to the components inside and outside the IIS system boundary.
- Any known connection information such as interfaces, ports, protocols, etc.

The IIS can present attack surfaces that are of specific interest to CSDS:

- External connections to the IIS – systems controlled by a separate authority outside the environment of operation may maintain connections that bypass security controls to allow for remote access or updates by vendors or service providers.

2.3.2.1.2 Performance

With the primary driver being to decrease the dwell time for cyber events, how the relevant data is offloaded, retrieved, or streamed from an IIS has a large impact on the efficacy of the functions in the CAC. These systems are very diverse; in some cases, they may have a high throughput, low latency optical fiber connection, while others may only have access to low-power wireless or direct serial connections to a host PC. In the worst case, the data may need to be offloaded manually to removable storage, in some cases as archaic as a 3.5” floppy disk. Due to this, it is necessary to address the basic parameters of the connection and its position on the network to understand what kinds of data can be provided and how frequently it can be sent.

In addition to the benefits to defining analytical models, knowing this information can be useful when building up CSDS capability in the environment of operation since there may be some systems that arbitrarily, rather than necessarily, have poor availability with sporadic or low-bandwidth connectivity. A better option may be available that is not used because the lower-quality connection has met the need. But as the need for low-latency observability into these systems increases, the team may need to take advantage of those options if they are available and its practicable to activate and maintain them.

There is also a certain degree of feedback loop here with the analytics, as once everything is up and running, the analysts may determine that a particular model will only really work with higher availability of a data element that is not currently available or isn't coming in often enough to be useful.

Following that, is it recommended that every component be connected at all times with enough bandwidth to stream all data in real-time? Looking from a purely cyber perspective, not necessarily. Securing that connection and minimizing the attack surface is also important and needs to be balanced against what can be gained from increasing the connectivity.

Sporadic or Continuous Connectivity

The connectivity of the system impacts the availability of data from the system, potentially increasing latency. Systems that have a continuous connection are more likely to be able to send more up-to-date information than those that are connected only sporadically. The distinction between continuous or sporadic connectivity is not related to connections that a system is making, only with the availability of connectivity with respect to that system.

With continuous availability of a connection, on-demand data streaming via long-lived connections are effective. Information can be pushed or requested from the system on-demand or periodically without the overhead of first establishing or waiting for connectivity.

Sporadically connected systems are not as well suited for automated on-demand data transfer. On-demand data requests initiated from outside the system can't be guaranteed. To gather the same amount of data from a sporadically connected system as one with a continuous connection, the system will need a larger local data cache to store the collected data while it waits for a connection to become available.

In the case where no meaningful networked connectivity is available, data can be transferred from the system via external mass storage devices such as external hard drives, secure digital cards, compact flash, or USB flash drives.

Network Edge Nodes

As with many terms in computing, this term means different things to different people; in this context, it refers to a property in a graph or tree-like structure where we are mapping connectivity or the ability to act upon other nodes. These are of interest because they offer natural points for collecting raw data and also can offer a way to distribute the load of processing that data. They also often act as gateways from one area of the data sphere to another.

Typically, an edge node is a device connected to a network with compute capacity and placed outside a traditional local or cloud datacenter. The term can be applied to many things, from IoT sensors and managed network devices to general-purpose computers. For example, in the case of the factory use case, a jumpbox that is dual-homed on two network segments may be considered an edge node – it is a compute-capable device and can act as an edge gateway between two networks. Edge nodes can help to collect and process data to reduce the load on the network and distribute compute requirements. Examples of devices that could be components of an IIS at an edge node:

- A jump-box acting as an edge gateway allowing for remote access to systems within a network segment.
- A managed network device such as a VPN server, network gateway, firewall, or proxy server.
- An IoT sensor that can process and pre-filter data.

Type of Connection

The physical network connection influences the data throughput and the reliability of the connection. For example, it is possible for serial bus connections to be low latency and high throughput (i.e., PCI Express) or significantly slower (i.e., RS232). In both cases, however, the topology and protocols are designed for a relatively small number of co-located nodes, making scaling difficult. Wired Ethernet connections are high-throughput, scalable, and pervasive; these will often be the best option.

Some systems may have hardware debug and in-system programming (ISP) or test access port (TAP) interfaces such as Joint Test Action Group (JTAG). These connections are used to flash firmware, debug, and validate programmable devices. The firmware can contain a bootloader with provisions for reprogramming using alternate interfaces such as RS232, I2C, and SPI. With more processing power and convergence toward SoC's, this increasingly can also include higher-level interfaces such as Ethernet or WiFi for over-the-air (OTA) updates.

Ideally, wired network connections will have the lowest latency and error rate with the highest throughput, but in some cases may not be feasible to implement. Depending on the use case, several competent wireless technologies exist, such as 802.11ax (capable of over 9 Gbit/sec data transfer). Efficient low-power wireless such as ZigBee M2M is capable of 250 kbps, typically has a shorter range, and has better low-level IO than WiFi transceivers.

Once a connection becomes available, the system can establish a connection to a data collection system, send the cached data, and free up cache space. The frequency at which connectivity is available, and the performance of the connection, when available, should be known. This information will help in selecting the types of data to be recorded, how the data is being stored, and the size of the local cache.

2.3.2.1.3 Security

Security considerations will vary wildly between different types of IIS and use cases (i.e., shop floor, safety-critical aircraft systems, IT/OT networks).

2.3.2.2 Data Acquisition Sensors (DAS)

From the perspective of the CAC, if a DAS didn't see it, it didn't happen.

Ideally, the domain stakeholder defines the CSDS-related goals, selects analytical functions that support those goals, generates a list of required input data, and determines placement of data acquisition sensors to capture that specific data. Constraints such as compute capacity, storage, bandwidth, and compliance and certification will impact the availability of data and placement of sensors. For more information on the DAS concept, reference the AAF-2.

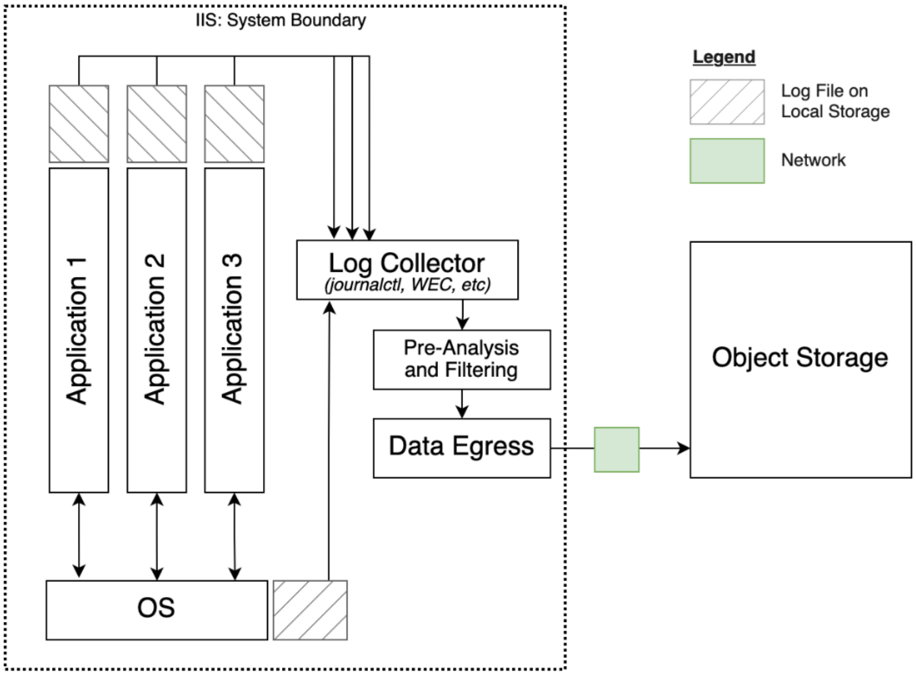


Figure 1: DAS Placed Within the IIS

2.3.2.2.1 Functions

Recommendations for sensor placement:

- Optimize the reliability, efficiency, and accuracy of the sensor. Key performance parameters must be defined to optimize the sensor, as the collected data is foundational to the CSDS AAF process.
- A particular piece of data may be available from many different locations within an environment of operation. As is practicable, select a point for data acquisition that is as close (considering the physical distance, network topology, number of hops, etc.) as possible to where the data was created. This ensures low acquisition latency and minimizes the chance of introducing error via factors such as re-transmission, translation, or data coercion.
- Sensors should have minimal impact on the systems they are measuring. Collecting data from a system should not impact its normal operation in any meaningful way. An acceptability threshold for impact on the system must be defined for factors such as compute, storage, and system downtime.

Considerations for Industrial Internet of Things (IIoT):

- The IIoT concept describes a network of industrial tools that can share data, interact, and collaborate in a distributed way. These benefits come with potential new attack surfaces; if implementing an IIoT approach for CSDS data collection, care should be taken to ensure that existing security controls remain effective and the benefits of existing network segmentation are not diminished.
- A shop floor implementing the IIoT concept can generate an abundance of data, necessitating sophisticated data management to maximize the value of the raw data. This will place heavier loads on the DAS pre-analysis and filtering functions and will require a higher level of performance for the host system.

2.3.2.2.2 Performance

The DAS is responsible for capturing data. The sensor can be a log aggregator like Prometheus, built-in system logging like journalctl, output from components within systems like Cisco CyberVision, or even an actual physical sensor.

Depending on specific data acquisition sensor implementation, the component elements of the interconnected individual systems will have varying levels of data processing requirements. If a system is recording logs in a human-readable, string-heavy format that has few or no machine-readable fields, relevant data must be parsed, reduced, and refined from the raw data before it can be useful to analytics tools. This data-processing task can be placed at any point on, or distributed across the aggregate data pipeline from the IIS components (edge processing) to the CAC (central processing).

Examples of processing tasks that a data acquisition sensor might perform:

- Log formatting when being recorded
- Maintaining data stream from sensors to local storage
- Processing data to extract and sanitize data
- Transforming data from one format to another
- Filtering data for CSDS requirements
- Metadata collection and processing

2.3.2.2.3 Security

Security is a concern as the sensor may have low level access to a lot of information, some of which may be sensitive. Depending on the connectivity, the sensor may have to place the data in local storage for a period of time waiting to offload; this should be encrypted in the case of sensitive data. Steps should be taken to protect any data transmitted or stored by the DAS. Any collected data that is identified as critical should be encrypted at rest and in transit.

2.3.2.3 Local Storage

The local storage component in the AAF is meant to act as an ephemeral point of storage for data – it should be considered volatile storage (Figure 2).

- Data should be moved from local storage as quickly as possible to a more centralized, secured, and reliable data store.
- Data should be stored in machine-readable formats (JSON, XML, CBOR, etc) wherever possible.
- Data compression can be used to reduce storage capacity requirements (gzip, 7zip), and if executed on the in-memory data stream, can also ease storage throughput requirements.

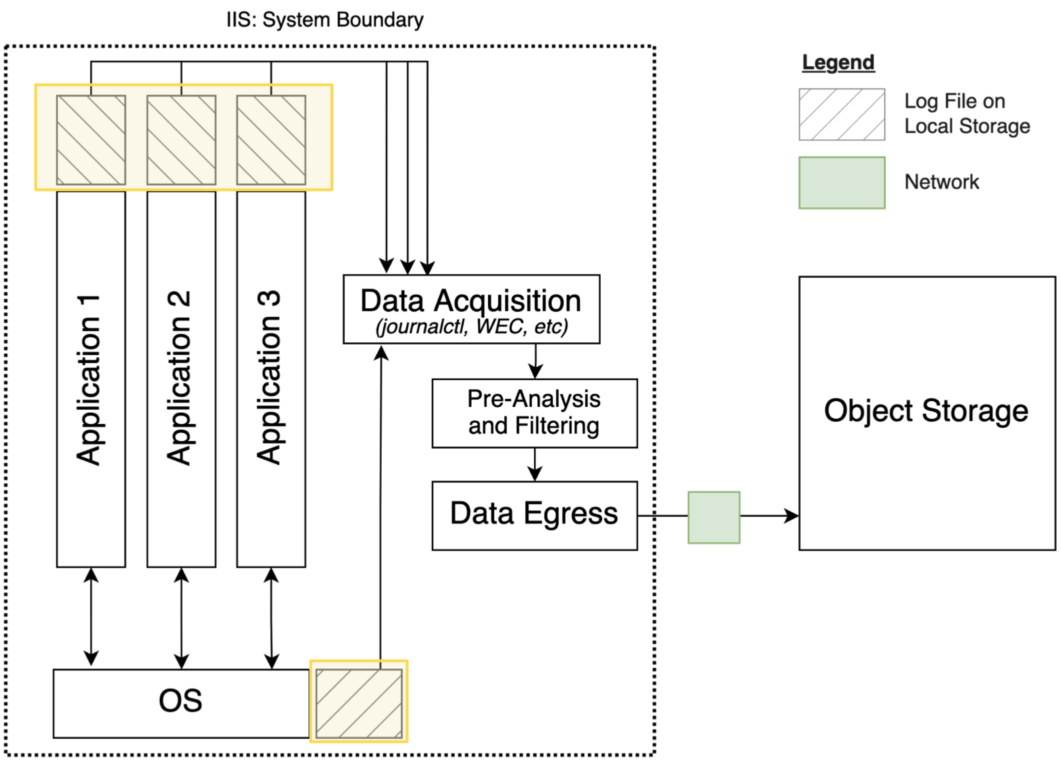


Figure 2: IIS with local storage highlighted

2.3.2.3.1 Functions

The local storage concept describes primitive storage locations that are located throughout the environment of operation. They can be embedded in an IIS (such as local disk drives and SD cards for data logging) or could be a storage-centric IIS (i.e., a storage array). While the local storage may analyze and process data with respect to its basic functions (filesystem journal, indexing, etc.), it does not analyze or process data with respect to the CSDS AAF data flow.

2.3.2.3.2 Performance

It is recommended to store the output of data acquisition sensors in a machine-readable format wherever possible. Some common machine-readable formats are discussed in Appendix A. The local storage component in the AAF is meant to act as an ephemeral point of storage for data – it should be considered volatile storage in that respect. Data should be moved from local storage as quickly as possible to a more centralized, secured, and reliable data store.

2.3.2.3.3 Security

Ideally, all data stored on local storage would be encrypted at rest, either in encrypted volumes or using whole-disk encryption. Realistically this is unlikely to be practicable when applying the CSDS AAF to existing systems that have strictly controlled configurations and limited resources, and so file-level encryption may be the only option available. Depending on the IIS, encryption of the data may become more critical, such as on systems that travel between sites or are exposed to the public. For more information on storage encryption, please reference NIST SP 800-111.

2.3.2.4 Data Egress Points

Data Egress Points (DEP) allow cyber-relevant data in local storage to be reliably retrieved and sent to Data-Stores (see Figure 3). They can be implemented in a multitude of ways depending on the constraints of the environment of operation.

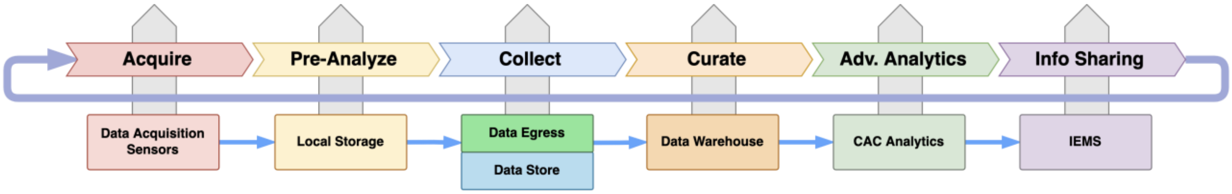


Figure 3: DEP placed against the AAF data lifecycle

2.3.2.4.1 Functions

The primary function of the Data Egress Point (DEP) is to allow cyber-relevant data in local storage (from data acquisition sensors) to be reliably retrieved and sent to Data-Stores. DEPs can be implemented in a multitude of ways depending on the constraints of the environment of operation (Figure 4). For background information on the DEP concept, please see AAF-2.

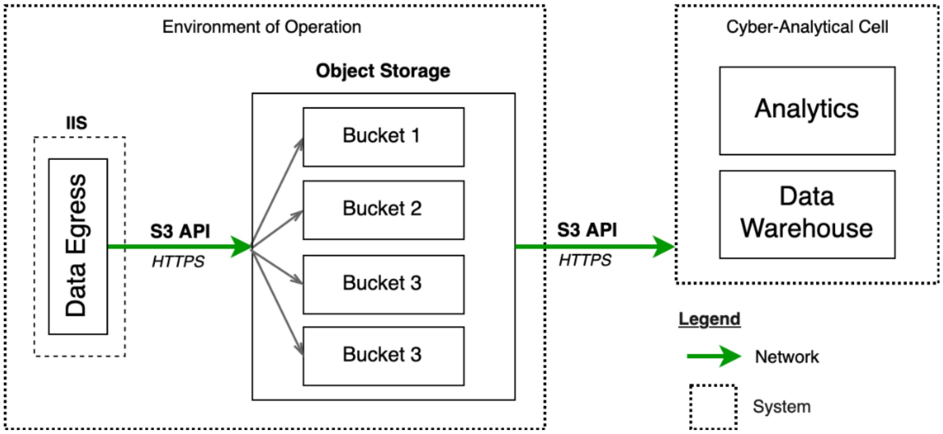


Figure 4: Data Egress from IIS to a Data-Store Implemented with S3 Object Storage

2.3.2.4.2 Performance

Metrics should be developed to measure the performance of the DEP against its primary function; by definition, these metrics should describe how well the data is being collected from local storage and sent to the Data-Store(s). Some guidance on general factors that can impact performance are listed below:

Is collection of the data from local storage manual or automatic?

- Automatic collection means that the data collected from sensors is automatically fed into the Data-Stores with no action taken by a human operator. This can happen continuously or on a set schedule and requires a data path between local storage and the Data-Store with continuous availability. Ideally, continuous data egress can yield the lowest latency and error rate when measured between when/where the data is collected and when it enters the Data-Store. Factors such as the availability of the DAS, the throughput of the connection(s) involved, and link latency may impact performance. When a continuous collection is not feasible or effective, automatic periodic collection can be implemented where data egress occurs on a fixed schedule designed around the constraints of accessing the local storage and transmitting the data to the Data-Store(s).
- Manual collection means that the data collected from the sensors is stored in local storage and then must manually be transferred by a human operator into the Data-Store. Manual collection can be done remotely, i.e., using remote data transfer with protocols such as FTP and HTTP, or via network filesystem shares such as NFS and SMB. Manual collection can also be done locally/physically using a temporary direct connection (Ethernet crossover or temporary patch, RS232, USB host), or with removable storage (USB Flash, SD Card, external disk drive).

What is the throughput and latency of the data path?

- The throughput and latency of the data path between the local storage and the Data-Store has an impact on the availability of the data and must be balanced against the type and size of data being collected (ultimately feeding into the concept of data velocity defined in AAF-2). Sensors may store data in a raw uncompressed format. In this case, implementing continuous automatic data egress may not be possible without taking additional action to reduce the data size (i.e., compression and filtering). Different compression techniques may be used to reduce the data size of the data even further (see Appendix A & B).

What is the priority of collecting the data with respect to the CSDS goals?

- Will the data help determine if there is a cyber-event pending, an attack occurring, or identify if an incident/event was caused by cyber activity? It can be difficult to tell what data could be useful in answering these questions, however, if any data is identified as relevant, its retrieval should be prioritized. Minimizing response time to threats is a primary driver in the success of implementing CSDS. When implementing collection of high-priority data, performance metrics that can be used to define minimum thresholds for data latency should be defined.

- For systems that have a large sporadic amount of data that may overload the network, a Quality of Service (QoS) profile may be utilized on a managed router or switch. Network prioritization is a common technique to ensure that packets from critical components are delivered when the network is overloaded. While employing QoS can help mitigate periodic congestion, a continuously congested network should be re-evaluated.

2.3.2.4.3 Security

Transmission of data to the data-stores should be strictly controlled. Devices and systems capable of accessing the data-store should be limited to those designated as Data Egress Points (DEPs). The aggregate DEPs are the primary data feed into the CAC, and their performance has a direct impact on the efficacy of the CSDS functions.

The DEP presents an attractive target for threat actors as a convergence of data throughout the environment of operations. DEPs will naturally be saturated with more data than other devices within an IIS, increasing the risk that a security breach of the device will yield security-critical data. DEPs also establish channels to exfiltrate data inside the EO to another location. General guidance on security and privacy controls that can be applied to DEPs can be found in NIST SP 800-53. At a minimum:

- All network communications to a DEP should use strong encryption. Selected encryption communication protocols should be standardized, such as Transport Layer Security (TLS). For guidance on selection, configuration, and use of TLS, see NIST 800-52.
- Key-based authentication should be used to control access to DEPs. All keys should be stored in a hardware security module (HSM) or appropriate secure key manager. Guidance on managing cryptographic keys can be found in NIST SP 800-57. Guidance for managing digital identity can be found in NIST SP 800-63.
- Authorization and access scopes should be restricted using least-privilege and least-functionality concepts.

For more general information on security guidelines for storage infrastructure, please see NIST SP 800-209.

2.3.2.5 Acquisition Modes (Removed)

2.3.2.6 Data Acquisition Sensor Placement (Moved)

See §2.3.2.2: Data Acquisition Sensors.

2.3.3 Stakeholder Data-Store Concept

The primary function of the data store is to reliably receive and store important CSDS-related data while it awaits extraction to the CAC data pipelines and/or data warehouse. Local storage is a limited resource and is not an archive. All data must be moved to a Data Store. There may be one or more data stores within an environment of operation (Figure 5). Network segmentation, data reduction and compliance requirements, network and device performance limitations impact Data Store locations. For more information on the Data-Store concept, please reference the AAF-2.

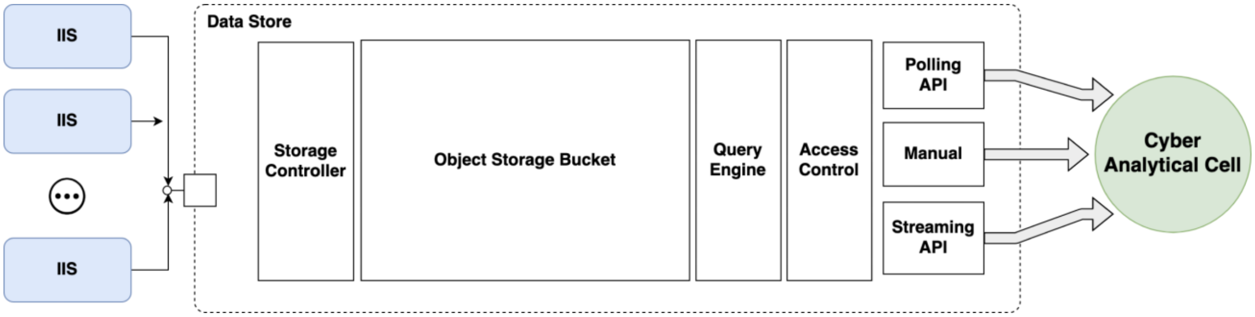


Figure 5: Diagram of an example data store implementation

2.3.3.1 Functions

As local storage is both a limited resource and not specifically designed to be a reliable data archive over time, the data collected there must at some point be moved to another more specialized component in the system – the Data-Store. There may be one or more data stores within an environment of operation depending on factors such as network segmentation, data reduction and compliance requirements, and network and device performance limitations. The primary function of the data store is to reliably receive and store important CSDS-related data while it awaits extraction to the CAC data pipelines and/or data warehouse.

2.3.3.2 Performance

In general, data storage follows a pattern of increased cost with performance that can jump orders of magnitude between storage technologies. This relationship and the location of various storage technologies with respect to performance (color coded) is illustrated in Figure 6: Data-Store Storage Performance Considerations. Stakeholders should verify that data storage devices, such as Data-Stores, can handle the data they are required to ingest. When possible, it is recommended that real data is used to create a set of requirements for data storage device performance vs. generic synthetic benchmarks (Figure 6). If the Data-Store is unable to meet the desired data velocity requirements, either the Data-Store performance can be increased, or the incoming data can be further pre-filtered or reduced by the Data Acquisition Sensors or further processed by the DEPs.

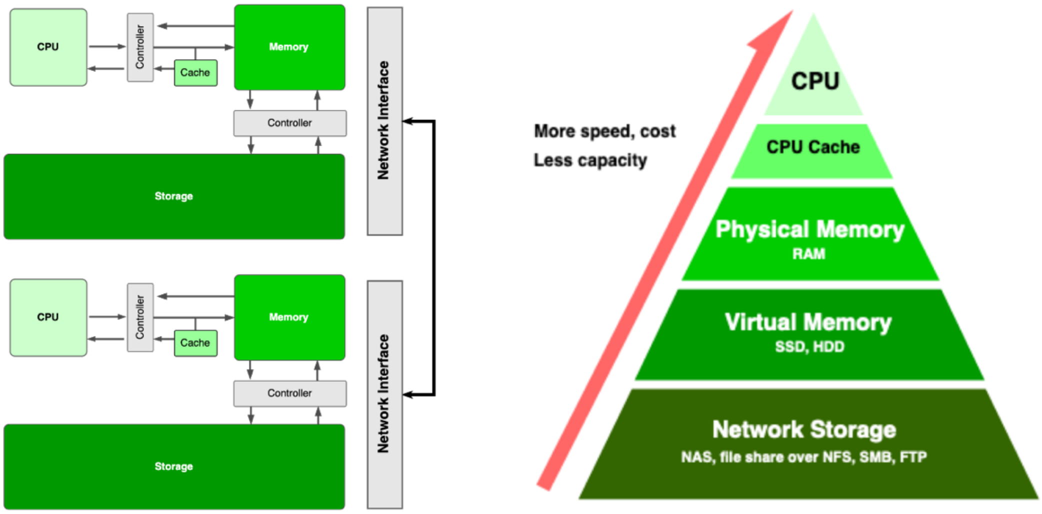


Figure 6: Data-Store Storage Performance Considerations

Each Data-Store will be required to handle the ingestion of data from one or more Data Egress Points (DEPs). There are several contributing factors in determining data velocity requirements for a Data-Store:

- The number of DEPs that are transmitting data to the Data-Store.
- The volume of data being sent from each DEP.
- The distribution of data transferred from a DEP over time. Is the data sent in fewer large packets or distributed across many smaller packets?
- The time-sensitivity of the data ingested. Is some or all data required to be processed within a certain amount of time to be useful?
- The life of the data in the Data-Store before it is extracted

Network-attached storage (NAS) or storage area network (SAN) solutions are popular and affordable data storage solutions that often implement redundant arrays of independent disks (RAID) to take advantage of the low cost of hard disk drives while parallelizing read and write operations across drives to maintain high data throughput (data striping and mirroring). An efficient and reliable middle ground would be a RAID-5 configuration, striping data with parity across a minimum of three drives.

Data throughput and redundancy can also be achieved through hybrid hardware and software-based storage solutions, such as Red Hat's Ceph Storage. This approach allows for fine-tuned control of the storage configuration to the needs of the use case at the expense of added complexity.

2.3.3.3 Security

The data store is an attractive target for threat actors as a centralized location of information and a nexus of communication between multiple IIS within the environment of operation. In general, the threats that must be considered are:

- Compromised or stolen credentials
- Unpatched vulnerabilities
- Scope or privilege escalation
- Ransomware
- Data loss via direct or indirect misconfiguration
- Man-in-the-middle attacks or network eavesdropping

A lapse in addressing these threats can result in:

- Data breaches, corruption, and unauthorized alteration of data
- Denial of service
- Compromised software related to data storage for the purposes of exploiting other systems on the network

For a comprehensive discussion of these threats, attack surfaces, and potential outcomes, please see *NIST SP 800-209: Security Guidelines for Storage Infrastructure,* §*3*.

2.3.4 Cyber Analytical Cells

Cyber Analytical Cells are represented by a collection of Human Analysts using software-based toolsets to perform analytics on data to produce cyber-analytical information within a highly secured networking environment. While it is ideal for CAC to be fully contained in one physical facility with Human Analysts working from dedicated workstations connected to the local LAN, it may not be possible or feasible. Human Analysts may be located anywhere in the world and may even work remotely. For more information on the CAC concept and multi-domain environment (Figure 7), please see *AAF-2*.

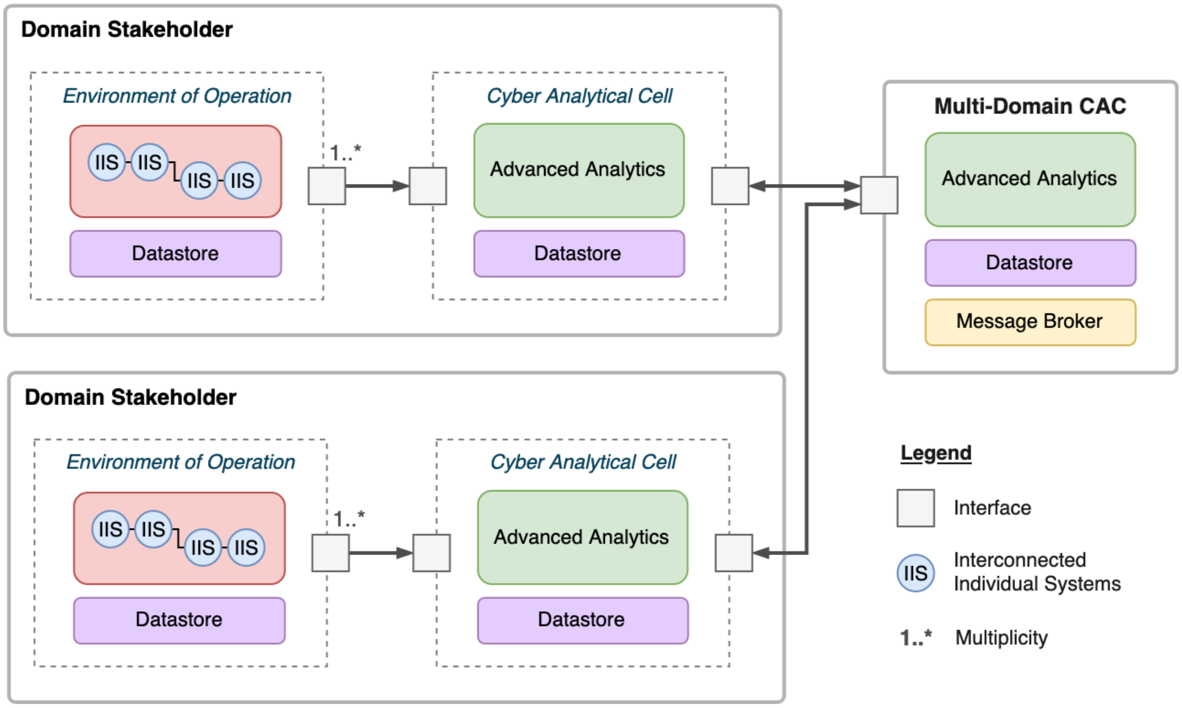


Figure 7: CAC and Multi-Domain CAC

Human Analysts in the CAC will need access to controlled software and data, such as the Data-Warehouse and Data-Store, when working remotely as well as in the office. Some security and operational challenges can be mitigated using secure thin clients to access centrally located and managed physical and virtual resources, maintaining a uniform configuration regardless of the analyst's physical location. Packaging and deploying some toolsets may require a local installation on each device used by analysts. Some toolsets will require the data they work with be stored locally; however, data retrieved from the Data-Warehouse should be managed carefully, as this can cause conflicts between different versions of the data. Recent events have necessitated a focus on remote work cybersecurity, generating a number of resources for information and best practices (i.e., Cybersecurity Risk and Mitigation Techniques During COVID-19 by Hossain, Riad, Shahriar, & Valero, 2021).

The CAC needs access to controlled software and data, such as the Data-Warehouse and Data-Store. Some security and operational challenges can be mitigated using secure thin clients:

- Access centrally located and managed physical and virtual resources.
- Maintains a uniform configuration regardless of the analyst's physical location.
- Easier to manage with central administration.

Packaging and deployment of some toolsets may require local installations and local datasets:

- Modification to local datasets should be tracked, and artifacts generated from modified datasets should be traceable back to that specific version.
- Using tools like Git LFS can help manage dataset versioning and workflow.

Access to certain toolsets or subsets of data may be limited to certain Human Analysts at a CAC. Reasons to restrict access may include:

- Access not required (least-privilege, limited scope)
- Lacking relevant training
- Compliance requirements
- Clearance level requirements

The most common method for restricting access to restrict access to networked resources has been perimeter-based and knowledge-based security, where an account and password can be used to pass through various perimeters of security. This strategy has been effective in the past, but has long been in retrograde with respect to the ever-evolving cyber-security landscape and increase in sophistication of threat actors. Ideally, a zero-trust architecture (ZTA) for shared resource access would be implemented, allowing for dynamic access control based on a wide range of variables. Please reference section 2.3.5.3 *Data Warehouse: Security*, for more information.

In a Cloud-Implemented CAC, the functions of the CAC can be implemented via a variety of cloud services that can generally be described by Platform as a Service (PaaS), Infrastructure as a Service (IaaS), or Software as a Service (SaaS). Please see NIST SP 800-145 for more information on definitions of cloud computing resources. Ideally, a cloud-based CAC would be hosted on services available from an established cloud provider such as Amazon Web Services (AWS) with GovCloud, Microsoft Azure, or Google Cloud Platform. With solutions such as AWS GovCloud, these options are certified to meet the needs of government customers that comply with the FedRAMP High baseline; the DOJ’s Criminal Justice Information Systems (CJIS) Security Policy; U.S. ITAR; EAR; Department of Defense (DoD) Cloud Computing Security Requirements Guide (SRG) for Impact Levels 2, 4 and 5; FIPS 140-2; IRS-1075; and other compliance regimes.

2.3.5 Data Warehouse

Data Warehouses provide localized long-term storage for Curated Data and Shareable Artifacts for CACs of both individual stakeholders and multi-domain users to enable a long-term analysis of patterns for multiple CSDS Use Cases. For more information on the Data Warehouse concept, please see *AAF-2*.

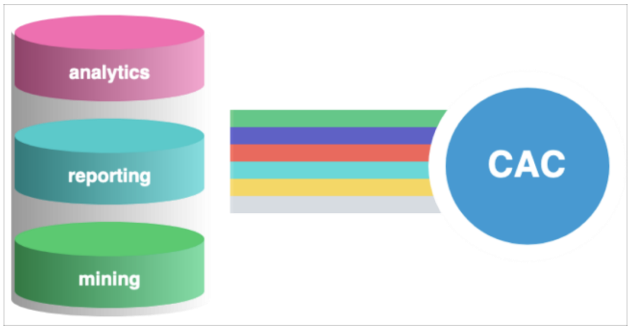


Figure 8: Three types of data stored in a Data Warehouse

When selecting or developing a data warehouse, these performance characteristics should be addressed:

- Determine necessary capacity and scaling plans.
- Ensure granular metadata and object versioning.
- Determine the average and peak number of requests to the API and any desired query processing.
- Identify the frequency and size of data transfer (ingress and egress).

The data warehouse will be a high-value target for threat actors; access should be scrictly controlled, and all accounts should use multi-factor authentication (MFA), preferably possession-based biometrics, FIDO2, etc. Where possible, apply Zero Trust Architecture (ZTA) principles and control access on per-request basis with a policy engine (PE).

2.3.5.1 Functionality

The Data-Warehouse functions as long-term data storage, and high storage capacity is required. The Data-Warehouse should expect millions of reads and writes per day as new data is generated and old data is referenced. The data should be stored using a scalable structure that supports granular metadata. Object storage systems are recommended because they are easily scalable, support granular metadata, and are best suited for relatively static data storage. Several solutions exist for object storage, with many supporting the AWS Simple Storage Service (S3) API. Metadata is critically important to the CSDS AAF to maintain the context of the collected data. In some tasks, the meta-data may be as useful as the data itself, such as in enforcing regulatory requirements for ITAR or PII.

2.3.5.2 Performance

Performance requirements for the Data-Store are driven by factors such as number of requests to the API, size and frequency of data transfer, data structure, and query processing. When designing or selecting a Data-Store, a benchmark can be used to define the level of performance of a particular solution and used to evaluate against the expected performance metrics. The major factors contributing to the performance of the Data-Warehouse are:

How often will the Data-Warehouse be accessed?

- Generally, the speed of storage media is proportional to the cost. For example, processor cache memory is many orders of magnitude faster and more expensive than secondary storage. This can also apply to specific groups of data within the warehouse.
- The number of users accessing the Data-Warehouse and the frequency of those accesses will have an impact on the system's ability to address each request in a timely manner. The frequency of access and the size of the transaction (which is discussed later) are both elements comprising the concept of data velocity, which is defined in Part 2.

What is the average size of data in transactions to and from the Data-Warehouse?

- Larger objects being written to or read from the Data-Warehouse will naturally take longer to transmit. The average size of transactions will impact the system's performance. Transaction size and frequency are elements comprising the data velocity concept, defined in Part 2.

Other than a stronger connection, low performance can be resolved with faster reads and writes at the local level. For extremely large storage arrays, a robust software-defined storage solution such as RedHat Ceph Storage might be appropriate, allowing the Data-Warehouse to scale up to any size necessary with fast, atomic transactions in a fault-resistant array.

2.3.5.3 Security

The Data-Warehouse is a priority target for threat actors looking to exfiltrate, corrupt, or deny access to data. For example, a compromised user account (stolen credentials, SIM swap attack) could allow a threat actor to collect or corrupt data, plant ransomware, or initiate a denial-of-service attack while evading detection. Possession-based authentication methods such as local biometrics, FIDO, WebAuthn are preferred over knowledge-based auth and credentialing such as SMS one-time-passwords (OTP) and knowledge-based authentication (KBA). Due to the placement of the Data-Warehouse in a non-public zone, distributed denial-of-service (DDoS) attacks are not likely; however, in cases where an attack surface is present to allow for a large DDoS, many mitigating services are available from cloud providers such as Microsoft, Google, and Amazon, as well as content delivery networks like Cloudflare and Akamai.

While perimeter-based security, network segmentation, device and service hardening, and intrusion detection systems are each important elements of implementing defense-in-depth across multiple layers of the organization, it is recommended that organizations also apply zero-trust architecture and concepts. The Federal Aviation Administration's (FAA) 2035 Vision for Air Traffic Management integrates the ZTA; instead of defending static network-based perimeters, zero trust focuses on securing information flow between individual assets and resources. For more information on ZTA, see NIST SP 800-207.

Access scopes for the Data-Warehouse should be strictly controlled and limited using a least-privilege strategy. Access to the Data-Warehouse should be controlled via a policy engine or enforcement point capable of implementing cryptographic controls like those used by the Data Egress Points. All transactions should be recorded in an audit log. See NIST publication SP 800-209 for more information on best security practices for storage structures.

2.3.6 Information Messaging System

2.3.6.1 Functions

The PubSub messaging pattern is advantageous for systems looking to decouple services producing information from those that process the information. It also eases the definition of asynchronous communication channels. The PubSub pattern can be applied to implementing event-based messaging in myriad ways from HTTP API-based client-server polling to decentralized embedded messaging libraries like ZeroMQ. Many cloud service providers offer a PubSub-based service for asynchronous message delivery with use cases ranging from big-data processing pipelines to mobile app push notifications.

When developing the information messaging system function for the CAC, the best fit for most organizations will be to implement a message broker. As the name suggests, a message broker acts as an intermediary between publishers and subscribers of information, controlling access, managing flow, translating between formats, and generally decoupling the applications that communicate through the broker.

2.3.6.2 Performance

When implementing the information messaging system in the CAC using a message broker, the primary performance considerations are centered around scaling the system as the rate and size of messages through the broker increases.

The utilization of load balancing techniques that are commonly used for distributing HTTP API calls across servers are not drop-in solutions, since the brokers need to accomplish some PubSub-centric goals such as ensuring that messages are received by a particular consumer at least once. For example, if placing two brokers behind a load balancer, when a client connects, it will connect to one of the two brokers. It is possible (and likely) that the next client to connect to the load balancer will be assigned to a different broker than the first client. In this situation, it would not be possible for either client to send a message to the other via a topic or queue. To implement this functionality, one approach is to ensure that the brokers are aware of each other and have a method to share messages from broker to broker. This is a capability found in many widely available message brokers, such as Apache ActiveMQ, that implement high-availability modes via primary/secondary failover, store-and-forward networks, and replicated message stores. For load distribution, store-and-forward and replicated message stores are particularly useful.

When implementing the CSDS AAF, reliable and high-performance data stores is a requirement for many components, such as the CAC's data warehouse and EO's data store(s). The same guidance for implementing those data stores (RAID, SAN, etc.) can be applied here in replicating message stores for message brokers. This allows for a set of brokers to mirror state across each instance.

A more flexible approach is to utilize built-in broker networking, such as is available in later versions of Apache ActiveMQ and Artemis, which effectively moves the job of the load-balancer inside the broker via either static lists or dynamic discovery. This can be combined with a set of rules in the broker that defines the desired topology (store-and-forward queues, mirroring, peering, etc.).

2.3.6.3 Security

An Information Messaging System should have the following characteristics:

- Authentication and authorization of each request to and from the system
- Strong encryption and signing for all messages
- Sandboxed environment for reading or parsing messages
- Anonymous messaging
    - Anonymity should be applied when a message is being sent between domains
    - Recipient should not learn who sent the message
    - Reports of messages sent should be available for administrators to search
- Enforced redaction of confidential/proprietary/sensitive information before a message is sent
    - Redaction of parts of data should not affect the original data – only the copy being sent

- Configurable restrictions on the types of data that can be shared (Data Governance)

There are many potential models for an Information Messaging System that would be appropriate. One such model is the Publisher/Subscriber model. For more information on the PubSub model, please see *Part 2 CSDS AAF – Technical Specification Document.*

2.4 CSDS AAF Data Architecture: The Data Perspective

2.4.1 Data Relevancy and the Data Sphere

Part 2 CSDS AAF – Technical Specification Document.

2.4.2 CSDS AAF Data Life Cycle Data Flows

For more information on the CSDS AAF Data Life-Cycle please see *AAF-2: Technical Specification Document.*

Data processing requirements for CSDS can be divided into two classes: system and user-oriented requirements. System-oriented data processing requirements are driven by the amount of information being processed by the system and tradeoffs between compute, storage, and bandwidth constraints. User-oriented data processing requirements are driven by user requirements such as service availability and response time, which in turn are generated or driven by business goals. The cyber-security related goals within the domain stakeholder's organization should be identified, along with how they relate to potential CSDS outputs. The overlap between the cybersecurity goals and CSDS outputs should be reflected in the objectives of the CSDS implementation within the organization. Examples of domain stakeholder cybersecurity-related objectives:

- Decrease time-to-detect
- Decrease dwell time
- Decrease recovery time
- Increase cyber-resiliency
- Reduce operational burden related to cybersecurity activities

Working from these objectives, a domain stakeholder can identify:

- The types of data that need to be collected
- The points within the environment of operation where the data can be acquired
- What type of sensors can capture the data with what level of fidelity and frequency
- Data pipelines for getting the acquired data back to the CAC for analysis

- The types of analysis that will provide actionable intelligence with respect to the objectives along with the necessary outputs of the CAC analytical functions
- New processes or process improvements needed to act on CAC outputs with respect to the objectives

2.4.2.1 Acquire

The goal of the Acquire phase is to capture data from the IIS that will be useful for CSDS efforts. The primary actor in the AAF for the acquire phase is the data acquisition sensor. When implementing the AAF, the types of data that need to be collected about the various IIS within the environment of operations should be defined as well as the locations where that data is available. Given this information, existing data telemetry and sensors can be identified, and any gaps in observability can be cataloged. For any observability gaps, existing sensors can be extended or duplicated to gain visibility to the data, or new sensors could be defined and placed. When changing existing telemetry gathering or sensing configurations, it is important to ensure that these changes can be accommodated with respect to compute capability (i.e., limited compute on embedded devices versus general-purpose computers), increased memory footprint, and storage speed and capacity requirements.

2.4.2.2 Pre-Analyze

Bandwidth for data egress from local storage is not unlimited, and some filtering and pre-analysis will be necessary to reduce and process the data. Depending on specific data acquisition sensor implementation, the component elements of the interconnected individual systems will have varying levels of system-oriented data processing requirements. If a system is recording logs in a human-readable, string-heavy format that has few or no machine-readable fields. In that case relevant data must be parsed, reduced, and refined from the raw data before it can be useful to analytics tools in the CAC. This data-processing task can be placed at any point or distributed across the aggregate data pipeline from the IIS components (edge processing) to the CAC (central processing). Examples of processing tasks in CSDS components:

- Log formatting when being recorded
- Maintaining data stream from sensors to local storage
- Processing data streams to extract and sanitize data
- Transforming data from one format to another
- Filtering data for CSDS requirements
- Maintaining and servicing data pipelines to CAC

2.4.2.3 Collect

During the Collect phase, the data may need to be restructured or partitioned to meet storage requirements and to allow for access and querying of the data. The selection of data store has a significant impact on the data; how it is accessed as well as what metadata can be stored or automatically generated. For example, many object storage solutions support generational versioning of the data, which can be useful in tracking changes to data over time. A natively time-based database (TSDB) can be particularly effective for storing time-series data. Examples of generic TSDB implementations are InfluxDB and Prometheus. Operations-specific time-series databases are often used in historian functions for operational process data.

2.4.2.4 Curate

The data should be stored using a structure that is scalable and supports metadata at a granular level. Object storage systems are recommended because they are easily scalable, support granular metadata, and are best suited for relatively static data storage, which fits this use case.

2.4.2.5 Advanced Analytics

Notifications could be sent via email, push notifications to apps, or published as events on specific topics on a message broker. Since email is relatively less secure than a dedicated end-to-end encrypted channel, the data included in the notification should be limited in accordance with the security of the channel being used.

Notifications should not be used as a method to transfer data in to or out of a data-store or CAC. The primary purpose is notifying a set of systems or individuals of an event, not to send all the data related to an event. The role of transferring the data related to an event should be restricted to communications channels available on the data-store and data warehouse.

2.4.2.6 Information Sharing

All data passing through the AAF should be marked with the data classification associated with, whether that be PII, ITAR, EAR, or FOUO. Data might be tagged with multiple restrictions, such as both PII and ITAR, or none. All systems that use this data must be aware of these restrictions, and any outputs they provide be marked accordingly based on the data that was used. Standards documents, such as NIST 800-122 in the case of PII, can provide general guidance on handling sensitive data.

3 Application of the CSDS AAF

CSDS AAF is proposed to facilitate the development and collaboration of CSDS techniques, tools, and processes in a systematic approach to assist cybersecurity analysts in answering three

(3) key questions for aviation architectures: 1) Is there a cyber-event pending? (Initial Foothold) 2) Is there an attack occurring now? (Network Propagation) 3) Was an incident/event caused by cyber activity? (Action on Objectives). These questions can be directly mapped to the Cybersecurity Unified Kill Chain model given in Figure 9. For more information, please see AAF-1.

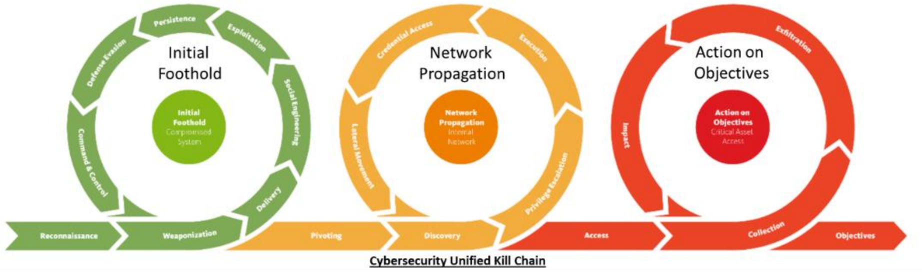


Figure 9: Cybersecurity Unified Kill Chain

One method of applying the CSDS AAF to a specific instantiation of an environment of operation (i.e. Aircraft OEM Factory) is to develop Use Cases and execute Analytical Exercises (AE) to explore and evaluate CSDS AAF application to those Use Cases via sets of scenarios.

> *CSDS AAF provides structure, key considerations, and criteria for applying CSDS methods to address the cybersecurity challenges through the development of practical aviation domain environment Use Cases. Use Cases represent the missions and goals associated with a Stakeholder's specific Environment of Operation. Use Cases will be used as the basis for implementing CSDS capabilities into the associated Environments of Operation.* [1]

3.1 CSDS AAF Analytical Exercise Concept

The CSDS AAF Analytical Exercise is a tabletop exercise where players respond to a scenario presented by facilitators outlining a hypothetical cyber-event carried out by a threat actor such as an advanced persistent threat (APT) group against an instantiation of the Stakeholder's specific Environment of Operation.[2] Facilitators present a series of facts to the players designed to appear innocuous, but in some cases representing a more serious issue. Some of the information presented should appear contradictory, or be presented as a distraction from the real problem.

[1] AAF-1 §4.4: Practical Applications of CSDS AAF.
[2] This format is strongly influenced by the CISA Cybersecurity Tabletop Exercise format.

The goal of the exercise is to bring the players together as a team to analyze a realistic scenario from a critical perspective, apply CSDS AAF to the scenario, and generate analysis and discussion that can be used to tailor the implementation as needed to match technical challenges and organizational goals.

To best execute a productive AE, facilitators and players should already be familiar with CSDS AAF and target Environment of Operation concepts and terms. Workshops and training should be provided as needed before execution of an AE.

3.1.1 Use Case Development

Use Cases can be mapped to the CSDS AAF for different Environments of Operations in the Aviation Ecosystem. Generically, a Use Case is a list of actions or event steps typically defining the interactions between an actor and a system to achieve a goal. The actor can be a human or another external system. The development process for a CSDS Use Case involves selecting and defining the Environment of Operation intended for application, identifying Threat Scenarios, and documenting the Use Case in the context of CSDS AAF. The following is a list of key potential items to include in a CSDS Use Case Document:

- Use Case ID – A globally unique ID that identifies this document
- The Use Case – A general description of the Use Case
- Description of the Environment of Operation of Interest – Refers to the Environments of Operation that belong to various Domain Stakeholders. The choice of environment, and the specific systems of focus, is typically driven by business risk considerations based on vulnerability data and threat Intel, such as perceived Primary Adversary Objective's. This should describe elements like the operational capabilities, people, processes, tools, input and outputs, and uses of the overall operational environment.
- Identification & Description of the IIS (IIS) – The various interconnected or networked systems that make up a given Environment of Operation. This will include the OT and IT interconnects, types of networks and interfaces, and types of data associated with the various systems that define the capabilities noted in the Environment of Operations described above. This should also include a systems security analysis for the identified areas of risk, considering known vulnerabilities and threats.
- CSDS Objectives – A list of CSDS Objectives that should be accomplished to a) Detect an impending Adversary Attack, b) Detect an ongoing Adversary Attack, c) provide most attack mitigation strategies/actions after an Adversary Attack.
- Threat Actors – Threat actors, their motivation, and possible attack surface.

- Threat Scenarios – A list of scenarios representing the goals of particular threat actors.
- Location of Available Relevant Data – A list of locations where available Relevant Data may be extracted from.
- Permitted Shared Artifacts – A list of permitted artifacts that can be shared with Multi-Domain CAC for the Use-Case as well as a Data Specification on the required and optional fields. This may also include reports and file formats.

The detailed steps or items that need to be considered in building out a Use Case include:

1. Select the target Environment of Operation (or sub-set thereof) for the Use Case
    a. Identify the systems configurations for the Use Case, and their associated intended uses.
    b. Identify the applicable network configurations and their operational use within the context of the Use Case.
    c. Locate and document the related IIS devices.
2. Define Threat Scenarios
    a. Identify the primary threat actor, motivation, and goal.
    b. Identify Threat Scenarios involving the threat actor
    c. Determine the risk levels of each Threat Scenario
    d. Investigate the threat actor's ongoing activities
    e. Determine ways to mitigate each Threat Scenario
3. Identify all available data for CSDS implementation and reduce to a set of relevant data.
4. Identify the cyber-relevant data from Step 3 and select data acquisition sensors:
    a. Evaluate the information type to be collected
    b. Research the sensor types that could be used to collect data
    c. Assess sensor capabilities for the collection of selected data
5. Identify the Data Extraction Process for each Data Acquisition Sensor
6. Identify or Define Data Stores
    a. Determine the appropriate Data Store Configuration
    b. Determine the Data Store element locations
    c. Consider local storage security and data retention policies
    d. Consider data management strategies to organize the acquired data
7. Identify or define Cyber Analytical Cells (CACs)
    a. Establish plans for a Security Operations Center (SOC) or other cybersecurity facilities for each Domain Stakeholder
    b. Identify tools for extracting data from Data Stores
    c. Identify tools for extracting data from Data Warehouses

    d. Identify tools for extracting data from Threat Intel Feeds
    e. Identify potential AI/ML algorithms for the data Pre-analysis, Curation, and Advanced Analytics phases that takes place within the CACs
8. Identify or define/determine Shareable Artifacts
    a. Determine what type of data will be considered Shareable Artifacts
    b. Ensure all Shareable Artifacts are sanitized and redacted from all confidential and sensitive data within the CACs
    c. Ensure Shareable Artifact are formatted following the data governance guidelines
9. Identify or define the Data Warehouse for CAC
    a. Consider long term storage that can process large amounts of queries a day
    b. Establish fault-tolerant data storage
    c. Store Shareable Artifacts in the Data Warehouse
10. Identify or define an Information Exchange Messaging System (IEMS) and how it will communicate with the data warehouse
11. Identify or define Multi-Domain CACs
    a. Multi-Domain Data Warehouse for long term data storage
    b. Data Governance guidelines
    c. IEMS to communicate with each Domain Stakeholder
    d. Collect Recommendations and Improvements

3.1.2 Scenario Development

A scenario must be developed prior to the AE for the selected Use Case. The scenario should be designed to execute against a specific instantiation of the Use Case in a specific Environment of Operation. An example of how to define such a scenario is outlined below:

1. Given the selected use case and specific scenario, identify a threat actor and goal.

2. Define background details on the threat actor: what tools do they use, what sectors do they usually target, what information is already known about this threat actor by security research groups, CISA, DHS, FBI, etc..

3. Define the attack that the threat actor will carry out, and detail/expand the associated cyber-kill-chain.

4. Identify challenges the attacker would encounter and develop countermeasures.

5. Fully define the initial capabilities and limitations of the malware to be used in the attack.

6. Establish the chain of events in the hypothetical scenario, assigning specific dates and times to each event. Adding detail in this step increases the fidelity of the AE.
7. Ensure the predicted impact of each event or action matches the goal defined in Step 1.
8. Define in a document or set of slides a presentation of the events established in Step 6 that is constrained to what the players could have known at each point along the event timeline. These events should be presented as a series of facts that may appear innocuous, but in some cases represent a more serious issue. Some of the information presented should appear contradictory, or be presented as a distraction from the real problem.

If player familiarity with CSDS AAF is high, the scenario may be developed specifically to exercise the organizations implementation. Otherwise, the AE can be used as a method to additionally familiarize the players with CSDS AAF, although at a minimum they should already be familiar with CSDS AAF and target Environment of Operation concepts and terms.

3.2 Analytical Exercise Example

This section presents an example of definition, execution, and analysis of a CSDS AAF Analytical Exercise. The CSDS AAF AE defined here was conducted at the FAA Florida NextGen Testbed (FTB) located at Embry-Riddle's Daytona Beach campus on February 6, 2023.

3.2.1 Use Case and Scenario Actors

This section provides a high-level overview of the scenario actors developed for the AE scenario, identifying the primary actors and the goals, tools, and methods of the threat actor.

The selected use case for the analytical exercise focused on a Third-Party Vendor performing a software update in the Aircraft OEM shop floor environment. As this exercise was executed as an example, all the elements in the tabletop scenario, such as actors, processes, and tools were created to act as analogs for their real-world counterparts. These model actors were used to represent actions that might be typical in the given scenario without drawing any specific conclusions on how the real-world counterpart would act. The actor stand-ins were defined as follows:

- **Nowell Commercial Aircraft** is a large aerospace corporation that designs, manufactures, and sells civil and military aerospace products worldwide.
- **AeonicAero** is an aerospace engineering manufacturer focusing on automation assembly systems.

- **APT94.** This is a state-sponsored group based on a conglomeration of details for real-world APT groups. They use sophisticated malware built in-house and adapted specifically for each target. The size of APT94's infrastructure implies a large organization with at least dozens, but potentially hundreds of human operators.

3.2.1.1 APT94 Details

APT94's goal is to compromise Nowell Aircraft's production facilities and create production delays on the Nowell Aircraft N320 line. The attack is timed to support an upcoming major sale of aircraft from the state's own aircraft company, Chrono Aircraft Corporation.

APT94 uses sophisticated malware built in-house and adapted specifically for each target. The malware is designed to evade detection by taking only passive actions and conducting low-risk recon until a set of predefined conditions are met. As this was a targeted attack, it is likely that the malware would have the capability to exploit vulnerabilities that might commonly be found in the Aircraft OEM environment. The model malware in this scenario is known as *MAL.Badwake*. The capabilities of this malware were based on profiles of the Sunburst malware developed by CISA, FBI, and various security research firms. The Badwake malware provides the following general capabilities:

- Remote command, control, communication (C3) with remote shell.
- Collect and upload system information. Run specified tasks, terminate processes.
- Download, read, write, move, delete, and execute files. Compute file hashes.
- Reboot the system and adjust process privileges.
- Can be configured to remain completely dormant for any period of time. Once active, it uses obfuscated blocklists consisting of hashed process and service names to identify analysis tools and antivirus software components running as processes, services, and drivers. If any of these tools are identified, the malware can remain dormant or take steps to evade detection.

3.2.2 Definition of the Aircraft OEM Factory Environment

This section describes the representative architecture of the Aircraft OEM factory environment as updated leading up to, and based on results from, the scenario presented at the AE.

The operational technology (OT) systems within the Aircraft OEM are generally divided into groups known as a *Work Cell*.

- There are at least tens of work cells in a large factory environment, each responsible for a specific task such as part manufacture, hole drilling, fastening, painting, and metrology.

- Automated ground vehicles (AGV) may be used to move larger parts from one work cell to another, and can also contain one or more work cells.
- The makeup of technology used in each work cell is highly non-homogenous, with a variety of different configurations depending on the task and the supplier/vendor responsible for the OT in the work cell.
- Components within a work cell are highly coupled, both functionally and with respect to interconnectivity.
- Each work cell is segregated from the enterprise network. Dual-homed PC's sit between the work cell network and the enterprise network. They do not bridge or provide a route between the two networks. In some cases, work cells may be connected to each other.
- While the work cell is operated by the Aircraft OEM, most of the systems within the work cell are built and maintained by a 3rd party vendor.

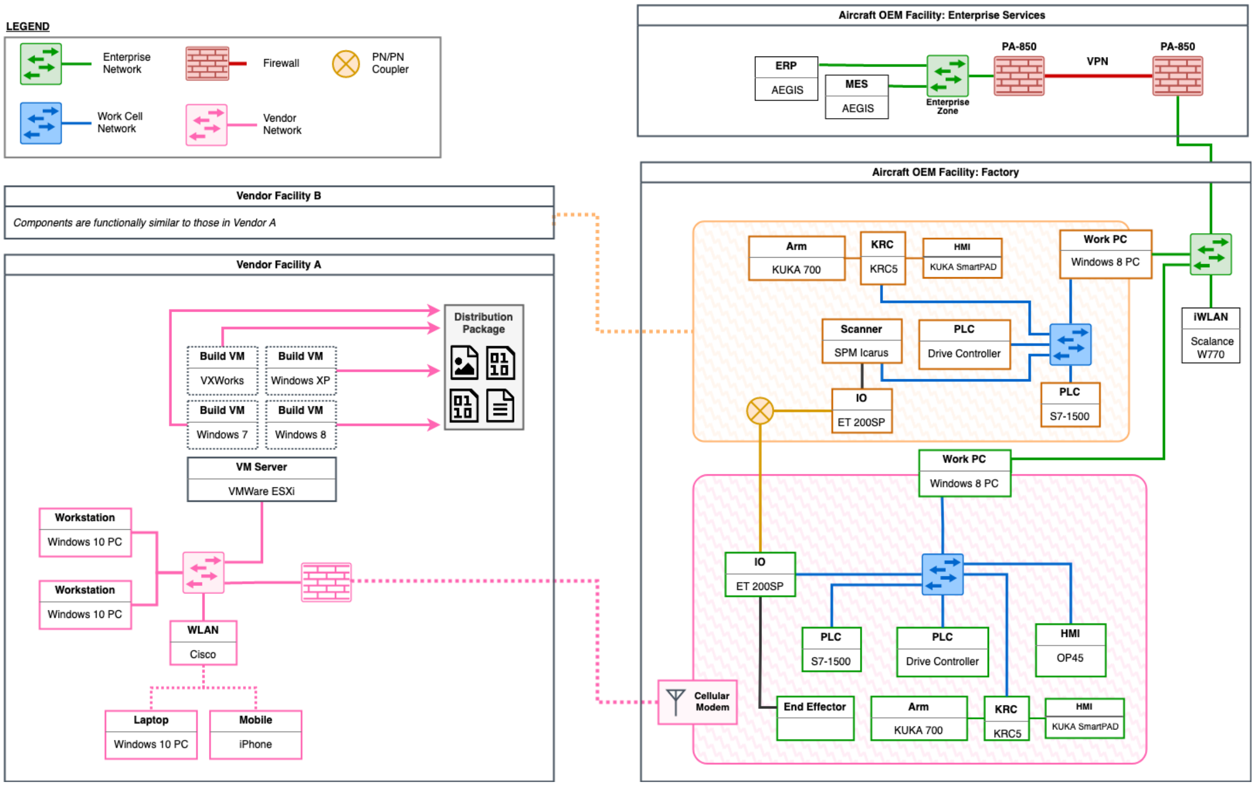


Figure 10: Aircraft OEM Factory Use Case Network Diagram

In Figure 10, the Aircraft OEM Facility (Nowell in the scenario) and associated systems is shown on the right and is depicted with two work cells with different configurations that are maintained by two separate vendors. The work cell in the pink box is responsible for composite part manufacture, while the work cell in the orange box is responsible for quality assurance (QA) and metrology tasks. On the left side of the diagram, two Vendors are shown, with Vendor A (AeonicAero in the scenario) being detailed.

3.2.3 Aircraft OEM Supply Chain Attack

This section outlines the details and mechanics of the specific attack on the functional Aircraft OEM environment that was developed for the AE scenario.

3.2.3.1 Summary

APT94 executes a spear-phishing attack on AeonicAero (Vendor) and leverages known vulnerabilities in the software Aeonic uses for team communication to steal credentials. Leveraging the stolen credentials, any unpatched and/or undiscovered flaws, or any misconfigured software within the environment, APT94 compromises the build chain for Aeonic's software development. They insert malicious code into the Aeonic binaries that can cause damage to Nowell's (Aircraft OEM) equipment and assemblies. These binaries are then distributed/installed at the Aircraft OEM's facility. Based on feedback from industry partners, the scenario progresses as the APT executes their attack and details the likely response from each actor.

3.2.3.2 Initial Attack Vector

APT94 targets the vendor using a spear-phishing campaign specifically designed to provide access to a known flaw in a popular team collaboration and communications software suite.

- Executing a spear-phishing attack and a zero-day vulnerability in Office 365 and Exchange Server, the threat actor can gain access to compromised machines for remote code execution (RCE).
- The attackers leverage this to access the team software flaw, allowing them to gather and use authentication tokens to compromise associated accounts and services.
- The attacker specifically targets users that have access to software development resources, and uses this capability to compromise the production build servers.

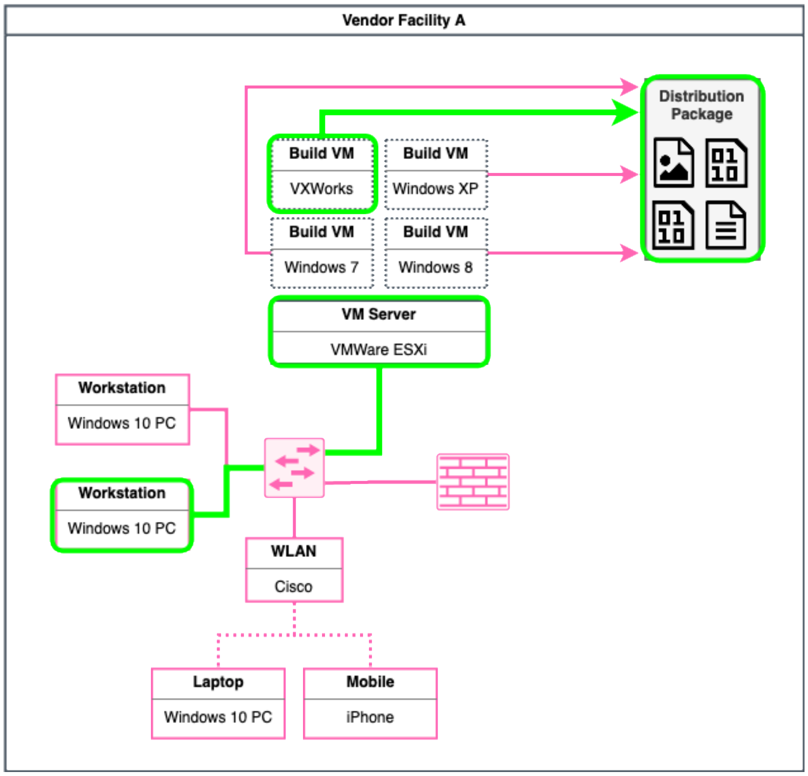


Figure 11: Compromised Systems and Network Paths at Vendor Facility A

Using the foothold on the workstation from the initial attack, the threat actor can take advantage of any misconfigured or vulnerable development or business workstations until they reach one with the ability to access the build server. The areas in Figure 11, highlighted in green, show the path of the attacker through the vendor network/build process. The goal of the attacker is to compromise the software build process such that malicious code can be injected into legitimate binaries without detection. By co-opting the normal build process, the attacker is able to maintain the established supply chain to the Aircraft OEM resulting in the distribution of malicious software through legitimate channels.

3.2.3.3 Lateral Movement from the Vendor to the Aircraft OEM

Updates of these binaries on systems at Nowell is a routine process, although frequency varies from one vendor and system to another. Due to the embedded and bespoke nature of the malware, early detection would be very difficult.

Ideally, the Aircraft OEM would deploy the binaries to a test environment before deploying them to a work cell. There are likely markers that the malware can use to actively determine that it is in a test environment to evade detection. The malware could also be passive, remaining completely dormant for a fixed period; this is risky for the attacker, as the longer the malware waits, the higher the chance that the initial breach at the vendor, AeonicAero, is detected.

Once the new binaries are loaded into the test environment and pass inspection, they are loaded into the production environment in one or more work cells. The malware can now begin recon and conduct malicious activities.

The malware has traditional IT based command and control (C2) that utilizes proven methods to provide a C2 channel while evading detection.

- The malware co-opts the AeonicAero Telemetry Protocol (ATP) to disguise network activity as normal AeonicAero system health traffic. While ATP is an imaginary protocol for this scenario, it is modeled after similar real-world application and system telemetry protocols.
- In addition to machines that use wired ethernet connections, there are many cases where industrial wireless LAN (iWLAN) devices are present within or adjacent to a work cell network. This can be leveraged to take advantage of the OT environment to use iWLAN devices to connect to other devices and networks that may be available over wireless protocols supported by the compromised device(s).

3.2.3.4 Impact on Aircraft OEM Environment of Operation

In this scenario, APT94 is targeting systems at Nowell to introduce defects into complex composite parts. For this scenario, it is assumed that APT94 could use Badwake to manipulate any or all of the following:

- Manipulate calibration settings for any robot in the work cell.
- Modify commands send to the robot from the PLCs.
- Change sensor values from the robot to the PLCs.
- Leverage any known (to the attacker; can be zero-day to the OEM or vendors) vulnerabilities on the robot, PLC, or directly adjacent systems to cause further damage.
- Move to the QA (orange) systems to cripple the ability for Nowell to detect defects, or to generate false positives.
- Compromise the Windows 8 PC and use it as a bridge to the enterprise networks.

3.2.4 Scenario Execution

This section outlines the scenario execution as presented at the AE. This is a different view of the information presented in 3.2.3, showing only what the Aircraft OEM would be expected to know and how they would be expected to react during the scenario.

The scenario starts with the vendor (AeonicAero) working with the Aircraft OEM (Nowell) to perform an update to one or more work cells. This update is required to fix a critical defect in control system software. This defect is not security related – it is a logic error in the control system code. Without the knowledge of the vendor or OEM, this update includes the Badwake malware. After over a week of testing the update in the test environment at the OEM, the update with malware is deployed to one or more work cells.

- **Day 1 - 37:** Workers on the shop floor use dual-homed PC's and associated OT systems in the work cells for daily tasks. There are no issues reported by the workers, and all parts and processes appear nominal. At this point the malware is collecting information and setting up a C2 path to the attacker.
- **Day 38:** QA systems detect defects in some parts. These defects are easily traced to a particular work cell. Due to the speed of manufacture and complexity of the parts, it is expected that a certain number of defects will naturally occur. The Aircraft OEM would be highly likely to not attribute any defects to malicious activity.
- **Day 70:** Analysts at the Aircraft OEM working at the enterprise level (perhaps with the enterprise resource planning software) note an elevated rate of defects in certain parts. The OEM investigates and determines that a set of work cells are producing parts with an elevated rate of defects. The OEM contacts the Vendor to investigate the issue. As the work cells are still producing mostly good parts, the OEM does not shut them down. The Aircraft OEM would be likely to not attribute any defects to malicious activity.
- **Day 71 - Ongoing:** Vendor and OEM work together in the test environment to identify the issue, but results are inconclusive. The vendor may reasonably conclude that there must be an unknown mechanical or software defect, and may take steps to mitigate it such as adjusting the process within the work cell to be more fault tolerant, replace hardware, reset settings, etc..
- **Day 78 - Ongoing:** One of the affected work cells has a major mechanical malfunction requiring an operator to trigger an emergency stop. There is damage to parts and equipment, and any dependent processes are also temporarily halted. The Aircraft OEM still may not attribute this incident to malicious activity.

In this scenario, after Day 78, joint analysis of the OEM and Vendor leads to the discovery that the same system operates differently in the test environment and the production environment despite the software and hardware being identical. At this point, malicious activity may be suspected. Further investigation into malicious activity could eventually find that the build environment at the Vendor had been compromised and was/is producing malicious binaries. Both teams could begin to remediate the situation at this point following their recovery processes.

3.2.5 Applying the AAF to the AE Scenario

As the familiarity with the CSDS AAF by the players was low, much of the discussion generated during and after the AE centered around how application of CSDS AAF might have impacted the chain of events that were presented. Would the vendor have detected a breach before the malicious binaries were installed at the Aircraft OEM? Would the Aircraft OEM have been able to source data from the Vendor into their own CSDS systems? What kinds of data would have provided an indicator that the Vendor was compromised? Would any data collected within the work cells indicate malicious activity? How would data collected on the Vendor-maintained resources within the Factory environment be sent to a Data Warehouse, and which Data Warehouse (OEM or Vendor) would it be sent to?

In this section, we will provide an example of how the AAF can be applied to the scenario described above to help answer the three (3) primary CSDS objectives: Is a cyber-event pending, is there an ongoing cyber-event, and what caused a cyber-event to happen?

3.2.5.1 Adding CSDS AAF System Elements to the Scenario

A modification of Figure 10 is presented in Figure 12 showing the Aircraft OEM Factory Use Case diagram augmented with AAF elements. These elements include data acquisition sensors, local storage (integrated with each sensor inside the IIS), datastores, data egress points, data warehouses, and the cyber-analytical cells (CACs). Blue dotted lines show data flow from sensors to data stores via data egress points. The CAC and data warehouse are represented with icons inside both the Vendor and Aircraft OEM Facilities.

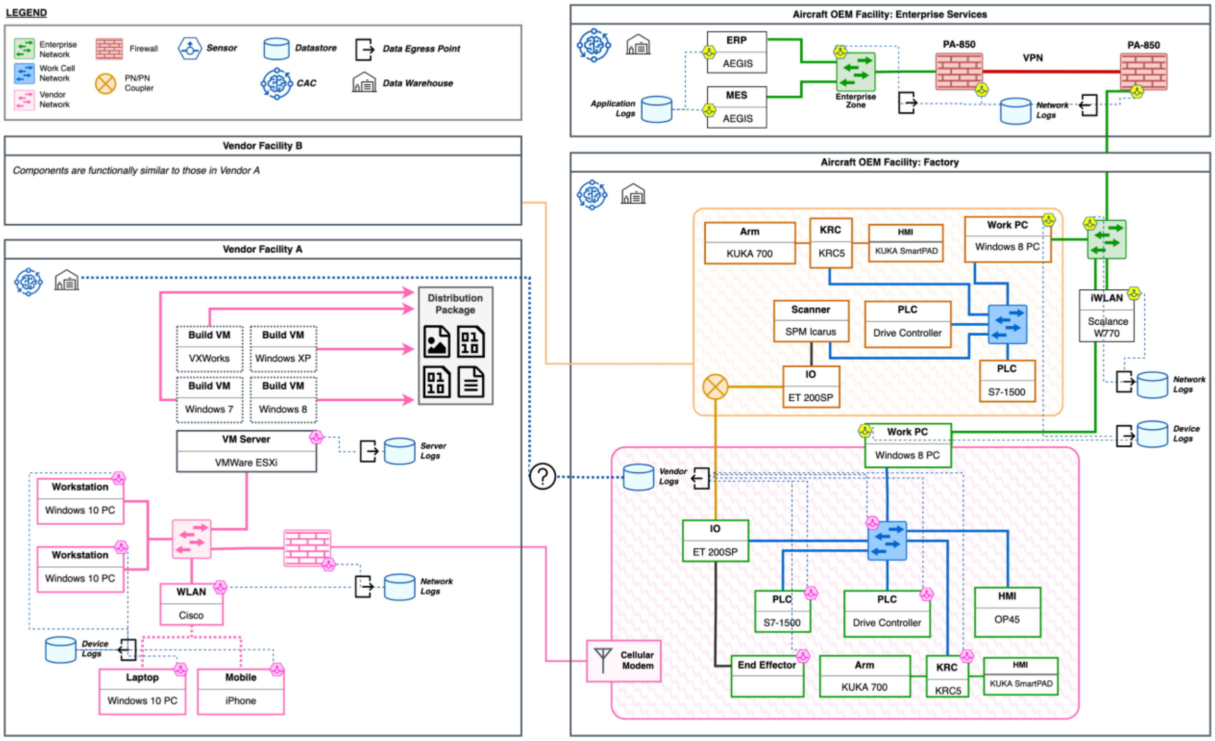


Figure 12: Aircraft OEM Factory Use Case With AAF Elements

In this scenario, the researchers will apply AAF Data Acquisition Sensors to locations where the sensing/observability already exists and identify what data is available or can be enabled via that existing sensor. The advantage of this approach over a top-down approach in this scenario is: Low overhead, low impact on current operations, and reduced time to a minimally viable CSDS AAF implementation. If this scenario was instead centered around a new vendor or system where the design was still being specified, it would be beneficial to define requirements for data acquisition from a top-down perspective.

3.2.5.1.1 Vendor Facility

In the AE scenario, the earliest opportunity for detection of the cyber-event is presented at the vendor. This is also the easiest place to detect the cyber-event, since the initial attack and the most lateral movement is conducted inside the vendor's environment. Inside the Vendor Facility, existing sensing/observability would reasonably include the following:

- **The end-user devices** such as desktop workstations, laptops, and mobile devices can collect system event logs (Windows Event Logs, managed iOS device telemetry, etc.), and if enrolled in an IT management suite, may also be able to provide rich information on user activity, installed applications, and attempted outbound network activity. This information would be stored locally before being sent to central data stores associated with whatever IT management suite is used within the organization. In the case of Windows workstations and servers, it is likely that the vendor would be making use of Active Directory for domain management. If the devices are domain-joined via Active Directory Domain Services (AD DS), then logins to those devices will be represented in the AD DS event log. Windows event logging and security auditing has been available since Windows NT 4.0, and can capture events for account logon and management activities, system events, privilege use, and global object access auditing. Similar logs are available on mobile platforms using mobile device management (MDM) software suites.

- **Servers**, like the VMWare ESXi server in this use case, will already have a plethora of system logs available, many of which may already be collected and sent to a centrally managed data store. By default the ESXi component log files are stored in a locally mounted */var/log* directory, divided into files for each major component such as authentication, system messages, shell logs, kernel logs, key provider service logs, and API logs.
- **VM Servers or Workstations** that are hosted on the ESXi server produce their own client logs as well as associated host logs from the hypervisor (ESXi in this case). The hypervisor would be able to very closely observe virtual machine state, providing highly detailed sensing on virtualized hardware. Many hypervisors, including the one in this example, also provide agent software that can run on whatever operating system (OS) that the VM is running to provide further integration with the hypervisor and provide additional logging.
- **Network devices** such as switches, firewalls, and routers collect logs locally and may also be able to send them to a network log server (such as *syslog*) or proprietary log server provided by the network device vendor (*Cisco IOS and NX-OS*).

For each sensor within the Vendor facility, a data egress point is required to efficiently and securely move the collected data from local storage into a data store. Depending on the sensor implementation, this may take the form of redirecting log output to a syslog server, collecting and syncing log files to a file server, or a fully integrated solution where the sensing, collecting, and storage are all provided by the same vendor (such as Cisco). In this case, logs from the team collaboration software and operating system on the Workstation (Windows 10 PC) could be made available to domain administrators at the vendor and automatically sent to various 3rd party log collection and aggregation products.

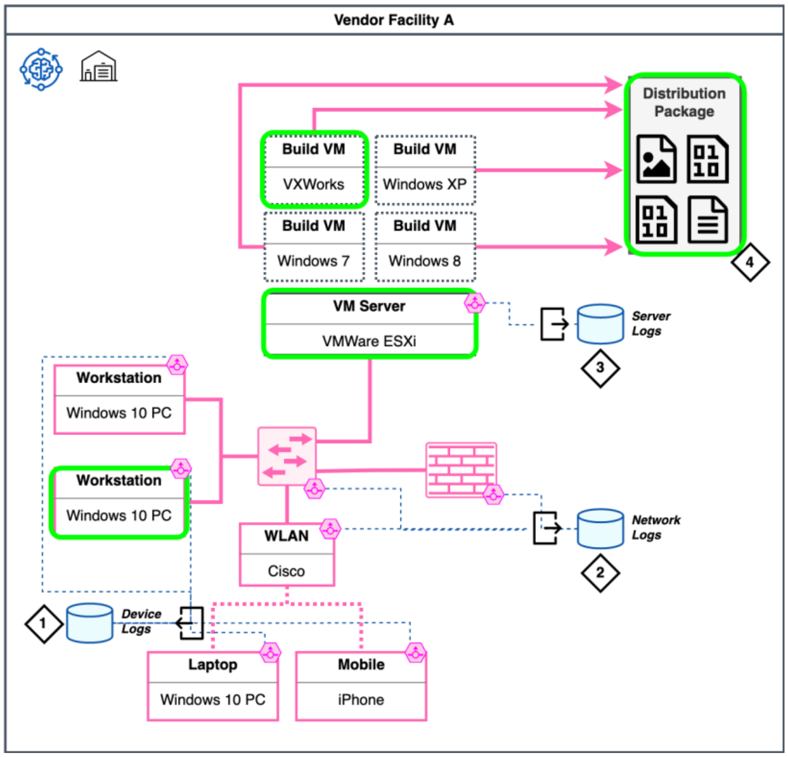


Figure 13: Overlay of AE Compromise with AAF Components in Vendor Facility

Data collected throughout the vendor's environment would be stored in the data warehouse in a predefined format that would have to at least capture the time, system, class of information/event, and metadata tags to be useful to analytical functions. The bulk of the analytical work for the CAC in this scenario is likely machine learning (ML) classifiers for identifying inputs that do not fall into one of the trained classifications, or perhaps do fall into a class for cyber-events. It is important that the data retrieved from the warehouse can be easily and efficiently represented in a form that resembles the form of the training data. Transfer and lifelong learning research has improved the capability of machine learning algorithms to deal with accommodating new knowledge and reuse of existing training on new datasets, but it is not a solved problem.

In Figure 13 four (4) opportunities in that a CSDS-enabled vendor could have detected the cyber-event presented in the AE have been labeled:

1. Device logs from the workstation running the team collaboration software. This was the initial vector for the threat actor, using a spear-phishing attack to take advantage of a vulnerability in the messaging software to collect credentials from the local data on the workstation. Attributes of the collected data that might have indicated a cyber-event is pending or in progress:

- In the case of a known unpatched defect or vulnerability, version information for the messaging software cross-referenced against automated threat feeds for known vulnerabilities could have alerted the vendor to the increased threat.
- The contents of the message, time of day, regularity with which the sender contacted the recipient in the past, and device logs verifying that the message was sent from a device registered to the sender could have indicated a spear-phishing attack.
- Once compromised, there would have been a window of time before the rest of the system could have been compromised where Windows logs to a central server may have indicated file access to a normally restricted or rarely accessed area of the filesystem (within the team collaboration software's internal file structure).

2. Network logs from the firewall or from a managed switch may have indicated abnormal activity. In order to move laterally to the build VM running on the VMWare ESXi server, the attacker would need to use the credentials to access either the VM or the server, both of which would have to transit the local network, even in the case of an automated script. In the case of active command-and-control (C2) from outside the local network, the data may have indicated abnormal activity at the firewall, although such activity may be very brief as the attacker would seek to disguise any such C2 activity.
3. Logs from the VMWare server. In the case where the attacker seeks to compromise the VMWare server itself in order to manipulate the VM's, logs from the server may indicate that a cyber-event is active in the case where access was gained via an exploit and not via stolen credentials. In the scenario at the AE, the access was gained via stolen credentials. In that case, the cyber-analytical systems would need to key off of other attributes such as abnormal access time, frequency of access, API's used, etc.. to identify that the access may not be legitimate. In the case where the attacker seeks to compromise just the build VM, the hypervisor and agent could provide good insight into the state of the VM and potentially identify malicious activity. In the case of stolen credentials, this would face similar challenges with identifying malicious activity as was the case in the VMWare server compromise.
4. The distribution package itself. This may require additional implementation on the part of the vendor, but it would also be possible to use security-focused scanning and testing of the end product (the executable binaries in this case). This extends from static analysis to running the code in isolated simulated environments. This would be part of the normal software development process for the vendor, the output of which would feed into the CAC.

3.2.5.1.2 Aircraft OEM

In this scenario the Aircraft OEM facility is divided into two environments: Enterprise Services and Factory. Systems inside Enterprise Services are controlled entirely by the Aircraft OEM and are already primarily concerned with data-related tasks. These systems include the Enterprise Resource Management (ERP) and Manufacturing Execution System (MES).

In contrast, many systems within the Factory environment are not controlled entirely by the Aircraft OEM. The diagram in Figure 14 illustrates two work cells within the Aircraft OEM's Factory environment, each with their own work cell networks and associated machines. The Aircraft OEM has the most control over the dual-homed Work PC's for both work cells. This PC will generally be running an OS image deployed by the OEM's IT. This level of control does not extend within the work cell. Each work cell is managed by a different 3rd party vendor. While the Aircraft OEM would be responsible for the day-to-day operation of both work cells, they have limited visibility into the internal workings of the systems within each work cell.

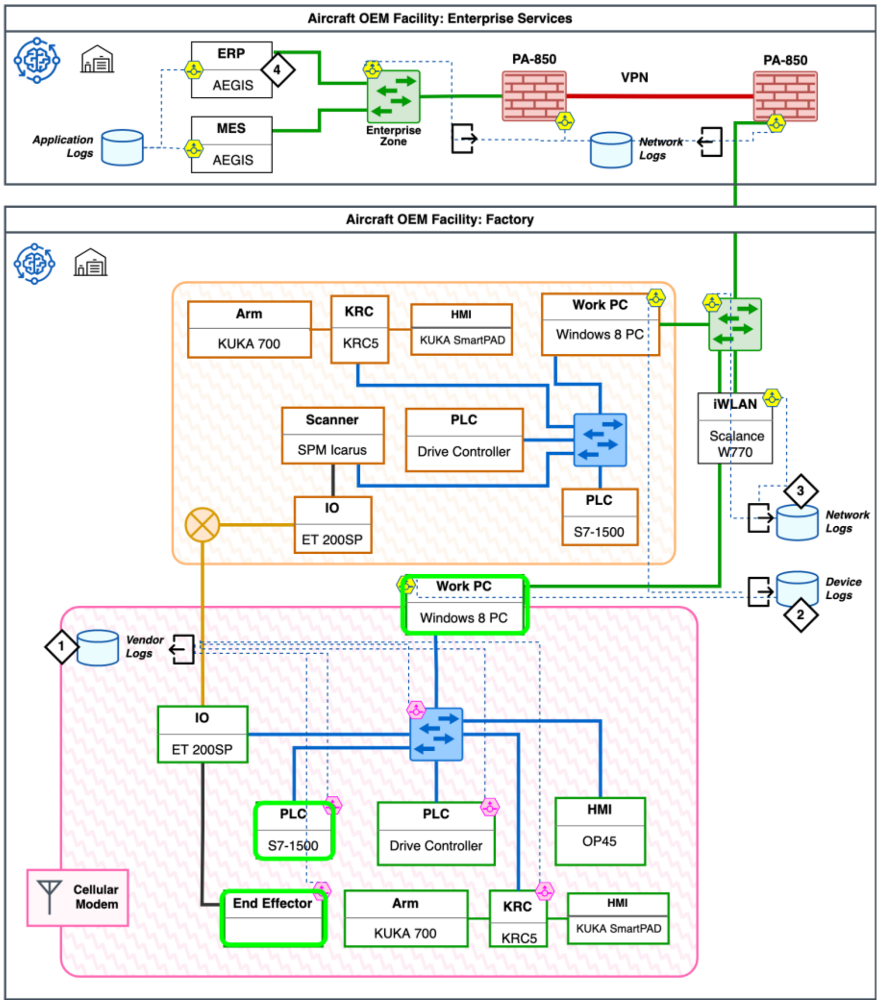


Figure 14: Overlay of AE Compromise with AAF Components for Aircraft OEM

As part of the manufacturing strategy, an Aircraft OEM may try to "black box" the system in a work cell as a whole, purchased, product. The control over the contents of this product are exercised through specification and feedback during the technical design of the system, and once the product is delivered, there is little opportunity to retrofit new capabilities. Furthermore, the vendor in most cases acts as an integrator, subcontracting work as needed for specific subsystems if they don't have the in-house expertise. This further lengthens the supply chain and dilutes control over the systems.

In Figure 14 the team has labeled four (4) opportunities that a CSDS-enabled Aircraft OEM could have detected the cyber-event presented in the AE:

1. Vendor device logs may indicate malicious or abnormal behavior. While the attackers may have compromised one of the PLC’s, the firmware on the end effector, and the Work PC, not all devices in the work cell are compromised. There are two problems with this data source: The first is that there is no way for data to leave the work cell other than manual transfer via the Work PC. Note that the data store for Vendor/device logs is inside the pink square. Fortunately this would be a violation of the AAF, as the framework clearly requires the data stores to be capable of providing their collected data to the data warehouse. In this case, either the data store itself must be located outside of the work cell network with a data egress point inside the work cell network, or a mechanism must be provided to allow the data to be offloaded from the data store in a secure and reliable way to the data warehouse. The second problem is that the entire work cell (from the cyber perspective) is outside of the day-to-day control of the Aircraft OEM, so the vendor would have to take action to collect and submit the data to the Aircraft OEM data warehouse within the factory facility, or the vendor would have to take action to collect the data and exfiltrate it back to the vendor facility, which would need a broadband connection from the work cell to the vendor traversing the OT and IT networks (more on this challenge is discussed in section 2.3.2.1).
2. The Aircraft OEM does have access to the device logs for the devices that they control that are associated with the work cell such as the Work PC and perhaps some OT hardware.
3. The network logs would be useful in this scenario to try to detect the C2 for the malware inside the work cell. Within the work cell, the access to the enterprise network is limited by design, so the Aircraft OEM should be able to develop a reliable profile of the types of communication that would be expected to come in and out of the Work PC. The attackers in the scenario co-opted an existing telemetry protocol that *was normal*, but it is possible

other contextual attributes such as time of day, frequency, data payload size, and send/receive patterns could be used by an analytical model to identify abnormal behavior.

4. Using the ERP Output as a sensor. During the AE, this is where the scenario officially had the Aircraft OEM catch a real abnormality in the form of increased defect rate. This was still not perceived as a cyber-event by the human analyst, but it is possible that an analytical model using maintenance data from the manufacturing systems combined with the defect rate could have, in conjunction with another data set (as described in 1-3 of this list), flagged the situation as an ongoing cyber-event.

Due to the “black box” nature of the work cell, it may be necessary for the Aircraft OEM to add their own sensors within the work cell to feed CSDS functions independently from the vendor.

3.2.5.1.3 Work Cell Supply Chain

The relationship between the OEM and vendors is such that the cyber-security of the vendor's systems, especially those within the vendor's own facility, is not seen as a technical problem for the Aircraft OEM, but instead is the responsibility of the vendor.

With each new attack vector used or surface exploited, there is an additional opportunity to detect a cyber-event, given that there is sensing in place to observe it and the systems available to recognize it as a threat. This is also a key benefit of defense-in-depth.

Competent cyber-related reporting requirements and controls must be in place for vendors and integrators. Going forward, more information sharing and transparency between the vendor and OEMs with respect to cyber-security data will be required to stay ahead of the accelerating threat of cyber-attacks. This cyber-security data should be treated as a critical component of the supply chain, and all entities involved should invest in developing capabilities to make use of it.

4 References

AA20-352A (2021, Apr). Advanced Persistent Threat Compromise of Government Agencies, Critical Infrastructure, and Private Sector Organizations. Retrieved from https://www.cisa.gov/news-events/cybersecurity-advisories/aa20-352a

Cisco Networking Knowledge Base (2019, Jan). How to configure logging in Cisco IOS. Retrieved from https://community.cisco.com/t5/networking-knowledge-base/how-to-configure-logging-in-cisco-ios/ta-p/3132434

NIST SP 800-111 (2007, Nov). Guide to Storage Encryption Technologies for End User Devices. Retrieved from https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-111.pdf

NIST SP 800-53 (2020, Dec). Security and Privacy Controls for Information Systems and Organizations. Retrieved from https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf

NIST SP 800-52 (2019, Aug). Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations. Retrieved from https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r2.pdf

NIST SP 800-57 (2020, May). Recommendation for Key Management: Part 1 – General. Retrieved from https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf

NIST SP 800-63 (2020, Mar). Digital Identity Guidelines. Retrieved from https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-3.pdf

NIST SP 800-209 (2020, Oct). Security Guidelines for Storage Infrastructure. Retrieved from https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-209.pdf

NIST SP 800-145 (2011, Sep). The NIST Definition of Cloud Computing. Retrieved from https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

NIST SP 800-207 (2020, Aug). Zero Trust Architecture. Retrieved from https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf

NIST SP 800-122 (2010, Apr). Guide to Protecting the Confidentiality of Personally Identifiable Information (PII). Retrieved from https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-122.pdf

NIST SP 800-82 (2015, Jun). Guide to Industrial Control Systems (ICS) Security. Retrieved from https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf

NIST IR 8183 (2020, Oct). Cybersecurity Framework Version 1.1 Manufacturing Profile. Retrieved from https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8183r1.pdf

VMWare vSphere Docs (2020, Jan). ESXi Log File Locations for vSphere 7.0. Retrieved from https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.monitoring.doc/GUID-832A2618-6B11-4A28-9672-93296DA931D0.html

A Data Formats

JSON

JSON is a very common data format due to its flexibility and balance of machine and human readability. JSON does not natively support schemas, although schemas can be implemented through use of third-party tools such as JsonSchema and OpenAPI. JSON provides support for generating dynamic data structures at run time, allowing for minimal pre-processing when encoding data input from many different formats. This could reduce integration time for log data sources that are not already machine-readable, as no formal schema is required, and the format is inherently resilient.

XML

XML represents data in a human-readable format that uses an element tree system to define the data being represented and maintain a structure. This structure can be verified using a Document Type Definition (DTD), which specifies the document's structure limitations. Specifying a DTD allows automated schema verification to ensure that the schema-defined structure has been satisfied. XML provides the ability to specify custom datatypes and properties via elements and attributes, allowing for complex type definition. This capability can result in XML-encoded data structures becoming cumbersome if schemas are over-defined or have been extended while maintaining backwards compatibility with earlier schemas.

CBOR

Concise Binary Object Representation (CBOR) is an extensible, binary-encoded data format that is similar to MessagePack. It also derives from JSON. CBOR was developed with a focus on small data sizes for the messages and especially for its encoders and decoders, allowing CBOR to be used even when limited storage and CPU throughput are issues.

BSON

BSON is a binary-encoded format derived from JSON. Where JSON focuses on ease of use and human-readability, BSON instead focuses on efficiency and performance of data transfer, access, and storage. BSON was originally designed as the primary storage format for MongoDB, a JSON document-based database. It includes data types not supported by JSON, such as ObjectID, Min key, UUID, and MD5. In MongoDB, these data types are used to support specific database operations, increasing performance across multi-node or replicated database structures. BSON provides additional features allowing for efficient modification without reserializing the data, reducing overhead for operations on properties in large data structures.

Protobuf

Protobuf is also a binary-encoded data format, but it does not derive from JSON. Protobuf was created by and is maintained by Google for the purpose of fast, structured binary encoding. The strict structure of Protobuf allows for implicit types to be understood when encoding and decoding data without those data types needing to be stored in the messages themselves. These types can be found in the definition of Protobuf, which sets forth strict rules for the placement of data when encoding, and by extension, where that data can be retrieved from when decoding. This design is in stark contrast to formats with loose definitions that store data types inline with the data, such as JSON and XML.

B Data Compression

Storage and network throughput requirements can be lowered at the expense of compute resources using data compression. The primary concern when choosing between them is the natural trade-off between smaller data sizes and computational efficiency -- compressing data into a smaller byte size without losing information takes time, so more heavily compressed data will require more processing time. In some cases, hardware-based compression may be available, mitigating much of the compute overhead on the general-purpose processor. If network throughput is constrained, an algorithm that heavily compresses the data would be preferred. Of secondary concern is the compatibility of the algorithm used. Widely supported compression algorithms are preferred versus bespoke compression due to security and overall efficiency across the CSDS AAF.

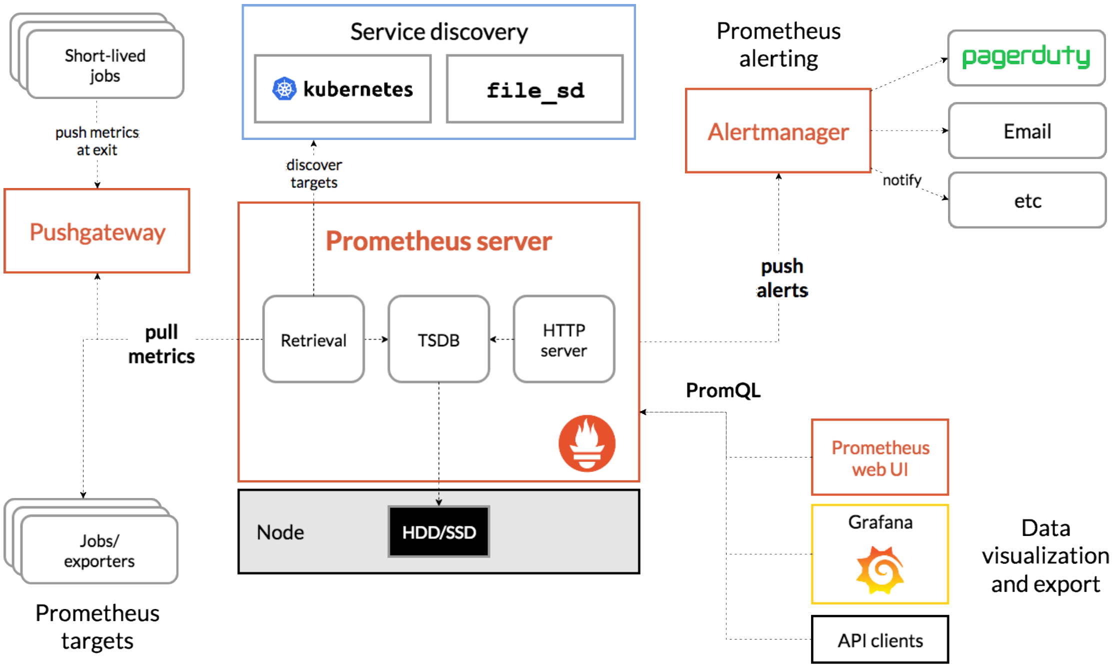
The gzip algorithm, often synonymous with the gzip file format (.gz), is one of the most common streamable compression algorithms available due to its balance between fast processing and efficient data minimization. Gzip is based on the DEFATE algorithm, which itself is a combination of LZ77 and Huffman coding (Deutsch P, 1996). To summarize the mechanics behind the lossless compression, gzip looks for patterns of elements that can be replaced with markers that take up less space than the patterns themselves. When decompressing, those markers are effectively replaced with the original pattern.

Format	**Size** $^{(KB)}$
XML	23.5
JSON	21.4
XML (minified)	19.6
JSON (minified)	12.3
MessagePack	11.6
CBOR	10.3
Protobuff	4.66
JSON (gzip)	2.45
XML (7zip)	2.19
JSON (7zip)	2.17

C COTS Data Acquisition Sensors

Prometheus

Prometheus is an open-source monitoring and alerting software that collects and stores metrics as time-series data with metadata in the form of key-value coded labels. Prometheus stores all data as time-series data streams, where each data stream is associated with a particular metric and set of labeled dimensions. The data labels allow for a dimensional data model to be used in conjunction with a query language to support filtering and aggregation. For example, a query can be used to identify, count, and extract meta-data fields from data streams on a specific target by specifying a metric name and a list of labels as key-value pairs. In this example, the researchers can define the metric as total requests to an HTTP API where the *method* used was "POST" and the *handler* was "/fwupdate". This can be encoded as a query and stored in the Prometheus server where it can execute over time to produce a metrics feed.



(Original image from: https://prometheus.io/docs/)

D NIST 8183r1 Mapping

Section	*Category*	*NISTIR 8183r1 Subcategory*
2.3.2.1	IIS	ID.AM-1..-6
2.3.2.2	Data Acquisition Sensors	PR.DS-1, 2, 4
2.3.2.3.3	Local Storage	PR.DS-3, 4
2.3.3	Data-Store	PR.MA-1
		PR.PT-2
		PR.DS-5, 6
		PR.IP-4, 6, 10
		800-82 REVISION 2 6.2.17.1 Virus and Malicious Code Detection
2.3.2.4.3	Data Egress Points	PR.DS-6
		PR.PT-3, 4
		DE.AE-1
		800-82 REVISION 2 Appendix E—ICS Security Capabilities and Tools