

The background of the page features a dark, textured design. It includes faint, glowing binary code (0s and 1s) and stylized circuit board traces that flow across the page, creating a high-tech, digital atmosphere.

DEPARTMENT OF THE AIR FORCE
APPLICATION PROGRAMMING INTERFACE

REFERENCE ARCHITECTURE

TABLE OF CONTENTS



Foreword	3
1. Overview / Executive Summary (AV-1)	4
1.1. Purpose	4
1.2. Scope	4
1.3. Principles	5
1.4. Assumptions	7
1.5. Outcomes for Warfighters	7
1.6. Architectural Products	8
2. Operational Activities Overview (OV-1)	9
3. Systems Overview (SV-4, SV-1, & SV-6)	10
3.1. Secure Data Access & Sharing	11
3.2. Interface Management	11
3.2.1. API Publication & Lifecycle Management	11
3.2.2. API Integration	12
3.3. Entity & Access Management	12
3.3.1. Entity Registration	12
3.3.2. Role Management	13
3.3.3. Credential Generation & Validation	13
3.4. Request Management	14
3.4.1. Request Flow Management	14
3.4.2. Application & Data Interfacing	15
4. Standards Overview (StdV-2)	16
Appendix: Acronyms	18



Foreword

The **Application Programming Interface (API) Reference Architecture** represents an important starting point in delivering the Department of the Air Force (DAF) API solution. To enable seamless collaboration and coordination across the DAF, Department of Defense (DoD), other federal organizations, and our allies and partners, the intent of the future API solution is to provide convergence across a vast application ecosystem that spans technologies and organizations. This initial architecture serves as a communication method between the DAF, our mission partners, the broader Department of Defense (DoD), and industry about the future strategic direction for a modern API ecosystem. **It is not intended to drive commercial product direction or formal proposals.**

Modern API implementations bridge the software development and data ecosystems to enable an **API-First** approach. This places APIs at the core of software design to drive interoperability, scalability, accessibility, and security. APIs will allow the DAF to integrate systems whether they are always connected to the enterprise or are in austere environments. This approach will allow the Air and Space Force to become more data-centric, supporting modern and agile systems to maintain the competitive advantage.

This architecture was built in collaboration with stakeholders across the Enterprise and DoD. It is aligned to the NIST Secure Software Development Framework and the NIST Software Supply Chain Security Guidance. We intend to incorporate further feedback from stakeholders into future versions of this API Reference Architecture.

If you have any feedback, please do not hesitate to let us know through the [Contact Form](#) on the SAF/CN website. Please select Chief Technology Officer (CTO) as the recipient and send us a message using the subject *API Reference Architecture & Roadmap*.

We look forward to continuing the API conversation.

MR. JAY BONCI

Chief Technology Officer

Department of the Air Force



1. Overview / Executive Summary (AV-1)

The **DAF Office of the Chief Information Officer's (SAF/CN)** mission is to provide the foundation for a secure, digital, and data-centric Air and Space Force by delivering decision advantage and shortening the kill chain. This API Reference Architecture supports success in SAF/CN's mission by improving secure data access and software interface sharing for the DAF. It is shaped by a commitment to an API-First approach that articulates APIs as the foundation of software design. Through this methodology, the architecture depicts common services and enterprise standards that will deliver increased security, accessibility, interoperability, and scalability of data access and sharing for our Airmen and Guardians as well as the commercial partners that support our missions.

This document depicts an overview of operational activities before describing the specific system functions, system features, data exchanges, and future standards required to improve secure data access and software interface sharing.

1.1. Purpose

The API Reference Architecture describes the critical elements the DAF will implement to improve secure data access and software interface sharing. It provides: **1) service design principles; 2) an understanding of the operational impact of an API solution; 3) definitions of foundational concepts and key technologies; 4) desired characteristics of resource flows between key technological interfaces; and 5) and future standards to shape the implementation of the overarching solution.**

The DAF must work towards a standardized way to communicate between software systems in real time. By providing architectural support at the enterprise level, software developers can be better equipped to develop modern applications. API-First design represents the idea that **APIs are the building blocks of software and encourage re-usability, appropriate granularity, and lifecycle management in mission system architectures.**

The API Reference Architecture documents the DAF's contribution to the visions for data and software interoperability at the DoD. This architecture considers the intent, outcomes, and frameworks described in three key strategy documents: *FY23-28 CIO Public Strategy*, *DoD Software Modernization Implementation Plan Summary (2023)*, and *DoD Data, Analytics, and Artificial Intelligence Adoption Strategy (2023)*. It will evolve based on feedback from the community, lessons learned during implementation, as well as any updates to DAF and DoD strategy.

Concepts included in this architecture also support the visions of other federal guidance and frameworks outside of the DoD. Specifically, it supports the vision laid out in Executive Order 14028, OMB M-22-18, and M-23-16 requirements. The architecture also aligns with the NIST Secure Software Development Framework, SP 800-218 and the NIST Software Supply Chain Security Guidance. It was also designed to incorporate OUSD R&E API Technical Guidance cybersecurity measures and best practices.

1.2. Scope

The API Reference Architecture describes how the DAF will implement an API-First methodology to improve secure data access and software interface sharing. Rather than immediately replacing existing solutions, the architecture describes a feature-complete end state that the DAF will deliver before converging existing solutions over time. This convergence will eventually support all software development and methods of data sharing across the DAF.



1.3. Principles

Seven service design principles shape the systems and system functions of this architecture. These principles are not intended to describe future requirements, but rather to begin a sketch of the features and functionality of the systems that will comprise a future enterprise API solution. All principles are subject to change based on feedback.

- **Principle 1. The API solution will establish modern architectures**
 - Principle 1.1. Design patterns will encourage modern, API-First designs that:
 - Force lifecycle management from the start;
 - Facilitate simplified management of multiple APIs per organization; and
 - Encourage creation and management of APIs to promote interoperability.
- **Principle 2. The API solution will be publisher- and developer-friendly**
 - Principle 2.1. Enterprise services will only relay API requests that conform to DAF API standards so that publishers will not have to manage API traffic
 - Principle 2.2. The architecture will facilitate interactions with published APIs through a DAF-supported Software Development Kit (SDK) and scale via enterprise gateways
 - Principle 2.3. Public documentation of API standards and conventions will help users navigate complex systems and promote code sharing and re-use
 - Principle 2.4. Publisher and developer participation in the ecosystem will be supported by pre-production API services as well as sample code
 - Principle 2.5. API publishers will avoid creating DAF-specific wrappers to existing APIs from third-party vendors, except to provide an independent access layer to DAF data
- **Principle 3. The API Architecture will be available at all classification levels**
 - Principle 3.1. APIs will work the same at each classification level, where appropriate, reducing the integration burden of delivering capabilities in highly secure environments
 - Principle 3.2. The API Architecture will enable high-side SDK add-ons to cover classified or in-development interfaces
 - Principle 3.3. Gateways will run as-a-service at all classification levels, where appropriate
 - Principle 3.4. The Enterprise API standards and associated services will support architectures designed to run in denied, degraded, intermittent, and limited (DDIL) environments
- **Principle 4. The API solution will have strong API standardization that addresses most use cases**
 - Principle 4.1. The API solution will initially support Representational State Transfer (REST) and JavaScript Object Notation (JSON)
 - Principle 4.2. API flow control and metadata standards will include message size and results pagination



- **Principle 5. The API solution will provide an enterprise gateway that assists in the secure scaling of API traffic**
 - Principle 5.1. Gateways will require additional security scrutiny due to their multi-tenant nature
 - Principle 5.2. The solution will ensure gateways only talk to endpoints, avoiding gateway to gateway communication relays
 - Principle 5.3. Gateways will perform message validation and coarse-grained access control before passing a request to the endpoint
 - Principle 5.4. Gateways should support response caching where requested by the API provider
 - Principle 5.5. The gateway will manage API traffic acting as a global load-balancer and reverse-proxy to protect API Endpoints
 - Principle 5.6. The solution will have API traffic logs available to cybersecurity providers and endpoint owners
 - Principle 5.7. Endpoints will be responsible to verify that requests do not circumvent the gateway
 - Principle 5.8. Clients will need assurance that they are talking to the gateway rather than a rogue intermediary by:
 - Validating the Domain Name System (DNS) entry for the gateway; and
 - Validating the cryptographic signatures on the API connection, likely via standardized checks in the SDK core library
- **Principle 6. The API solution will make all APIs available through a single SDK framework**
 - Principle 6.1. The DAF will support modern and appropriate major programming languages
 - Principle 6.2. The SDK style will be expressed in a format that matches the language it is targeting (e.g., matching casing and underscores)
 - Principle 6.3. Documentation will be available for raw protocols so developers can publish alternative SDKs
 - Principle 6.4. The SDK will be distributed wherever possible through language specific package management mechanisms (e.g., NPM, PIP)
 - Principle 6.5. The SDK will receive regular updates to directly reflect lifecycle events across the API Catalog
 - Principle 6.6. The SDK will have an independently versioned core library function that is maintained separately from API Catalog lifecycle events
 - Principle 6.7. The SDK will cover both participant-provided and internal APIs (e.g., Secure Token Service (STS), Identity Access Management (IAM))
 - Principle 6.8. The SDK will be a reference implementation for automated adherence to standards and policy



- **Principle 7. The API Gateway will govern request flow rather than content**

- Principle 7.2. There will be minimal API response and data format translation inside of the API Gateway to avoid creating a fragile delivery ecosystem
- Principle 7.3. The API Gateway will include rate limiting controls and provide request budgeting features
- Principle 7.4. The API Gateway will return appropriate error codes even when the endpoint does not return a well-timed, proper response

1.4. Assumptions

The assumptions listed below reflect the DAF's goal to integrate with DAF and DoD data, Enterprise Identity Credential and Access Management (ICAM), and Enterprise Zero Trust (ZT) efforts.

- Data that is served via APIs adheres to the **VAULTIS Framework** that ensures data is visible, accessible, understandable, linked, trustworthy, interoperable, and secure as well as the **Data Quality Dimensions set forth by the DoD**;
- Services described in the architecture are supported by the broader **DAF Zero Trust Architecture and are in part implemented within the DAF ICAM solution**;
- The implementation of this architecture will be a phased rollout of supporting services with pilot organizations working together to **test standards as they are proposed**; and
- This architecture must be supported by existing or near-term commercial products so that delivery of a solution can be built upon a robust ecosystem of technology providers.

1.5. Outcomes for Warfighters

- **Security** | Enterprise API Gateways will provide a standardized visibility and policy enforcement point for modern application traffic.
- **Accessibility** | Enterprise API services will connect end users and systems to data and software interfaces by facilitating access across missions and organizations.
- **Interoperability** | Enterprise API services will standardize and increase the ease of integration with our mission partners, other government entities, and industry providers.
- **Scalability** | Enterprise API standards and services will help API providers manage the volume of requests incurred by participating in a future, more interoperable environment.



1.6. Architectural Products

Product	Short Name	Description
Overview / Executive Summary	AV-1	Presents the purpose, scope, and subjects of an architecture effort
Operational Activities Overview	OV-1	Presents the concepts of operation of a described architecture
Systems Functionality Description	SV-4	The hierarchical structure of system activities and their resource flows
Systems Interface Description	SV-1	The identification of system resource flows and their composition
Systems Resource Flow Matrix	SV-6	The details of resource flows among systems; the activities performed; the resources exchanged; and the attributes (rules and measures) associated with these exchanges
Standards Forecast	StdV-2	Presents rules that will constrain activities and their performers in the future

Table 1: The Architectural Products table presents the different DODAF Viewpoints utilized in this architecture



2. Operational Activities Overview (OV-1)

The Operational Activities Overview (OV-1) depicts the integration of data from multiple mission systems into one software application using the DAF's API Architecture. The Systems Viewpoint discussed later in this architecture covers the specific systems included in the solution, expanding on the interaction displayed between API Gateway, API Catalog, and API Endpoint within the overall API request and response process.

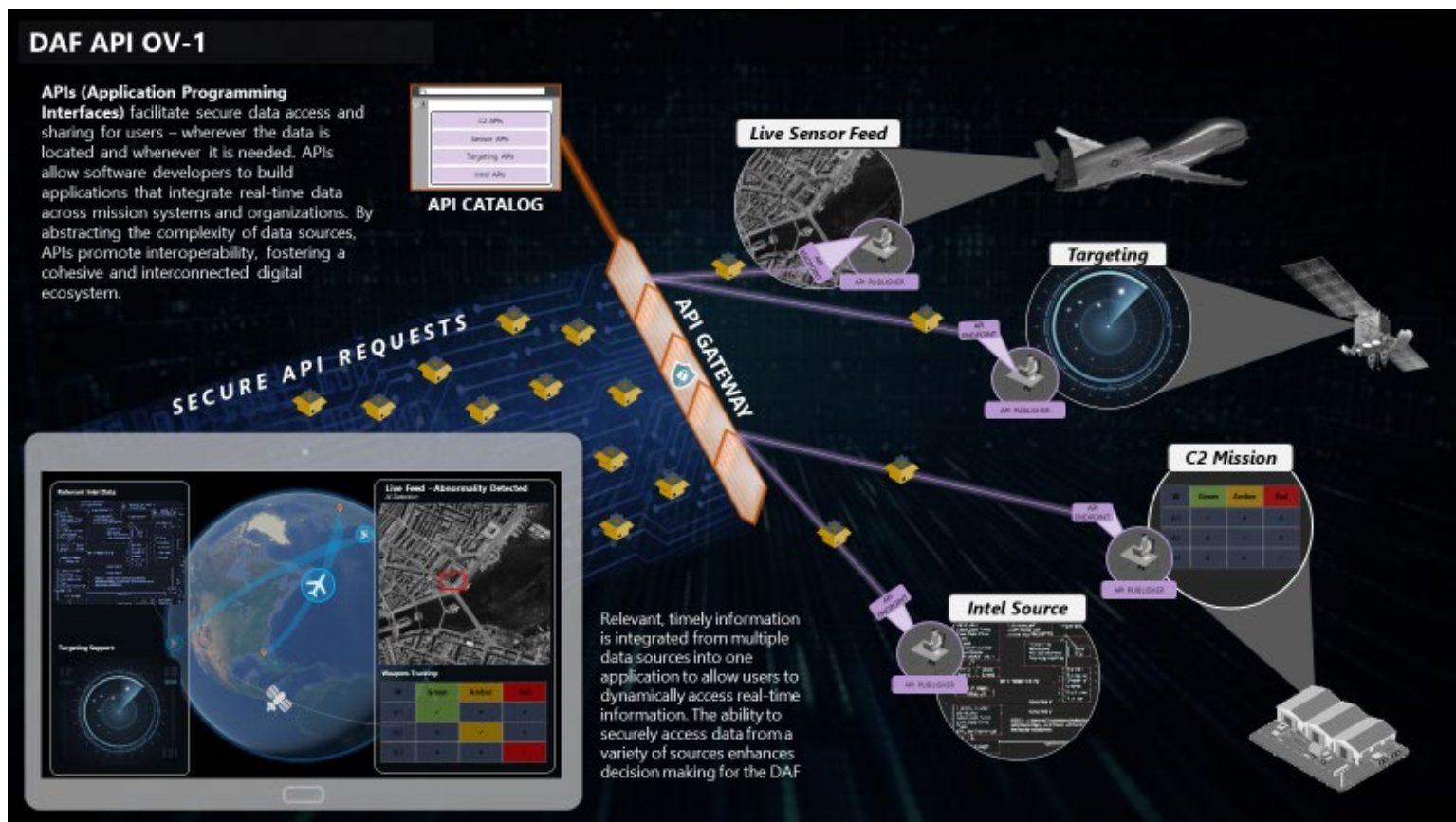


Figure 1: The OV-1 Operational Activities Overview presents the concepts of operation of a described architecture



3. Systems Overview (SV-4, SV-1, & SV-6)

A variety of technological systems and services are necessary to deliver a feature-complete API ecosystem. Figure 2 depicts the critical system functions for the ecosystem and Figure 3 further specifies the solution by mapping specific system interfaces. Finally, Table 2 details the data flows between the system interfaces found in Figure 3.

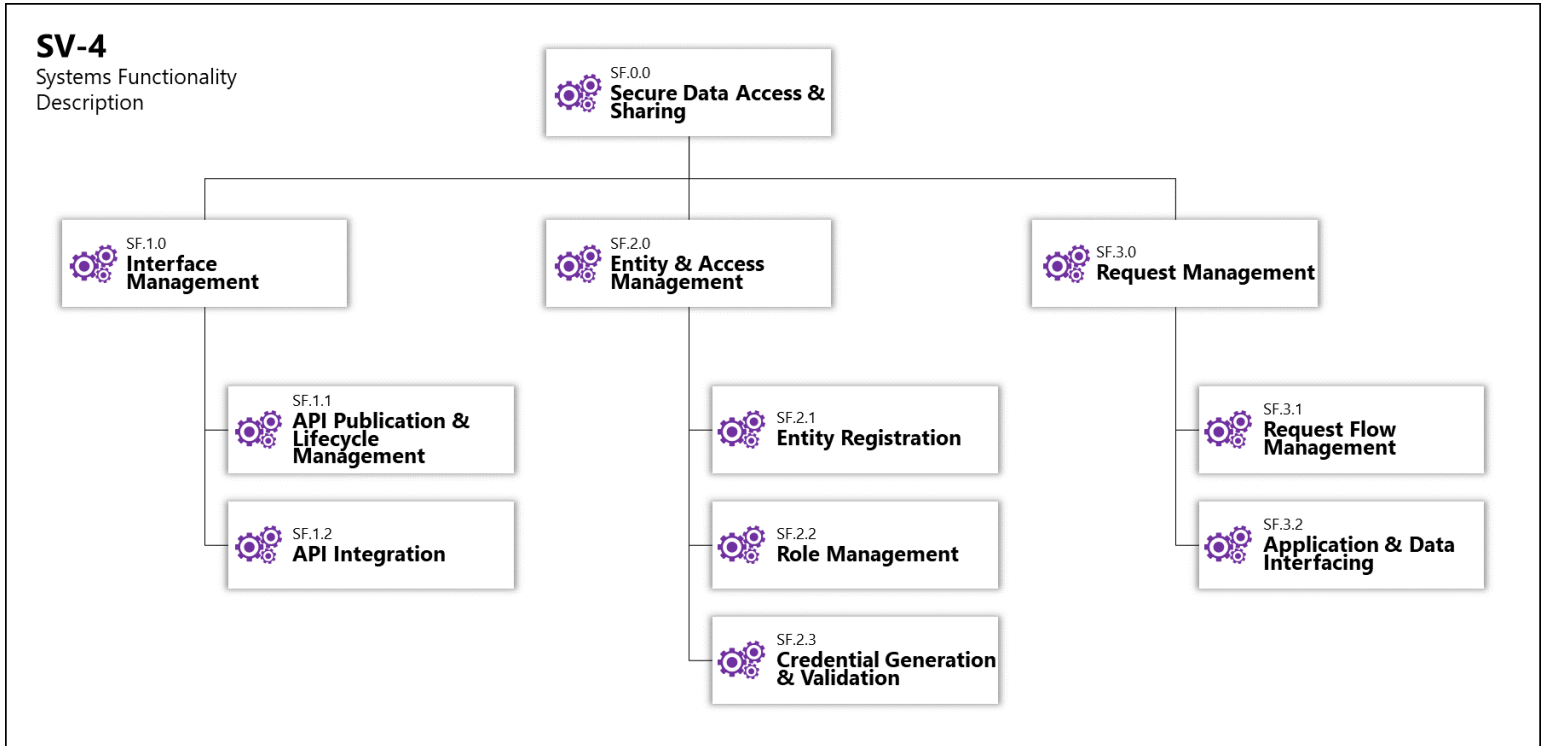


Figure 2: The SV-4 Systems Functionality Description shows the hierarchical structure of system activities and their resource flows

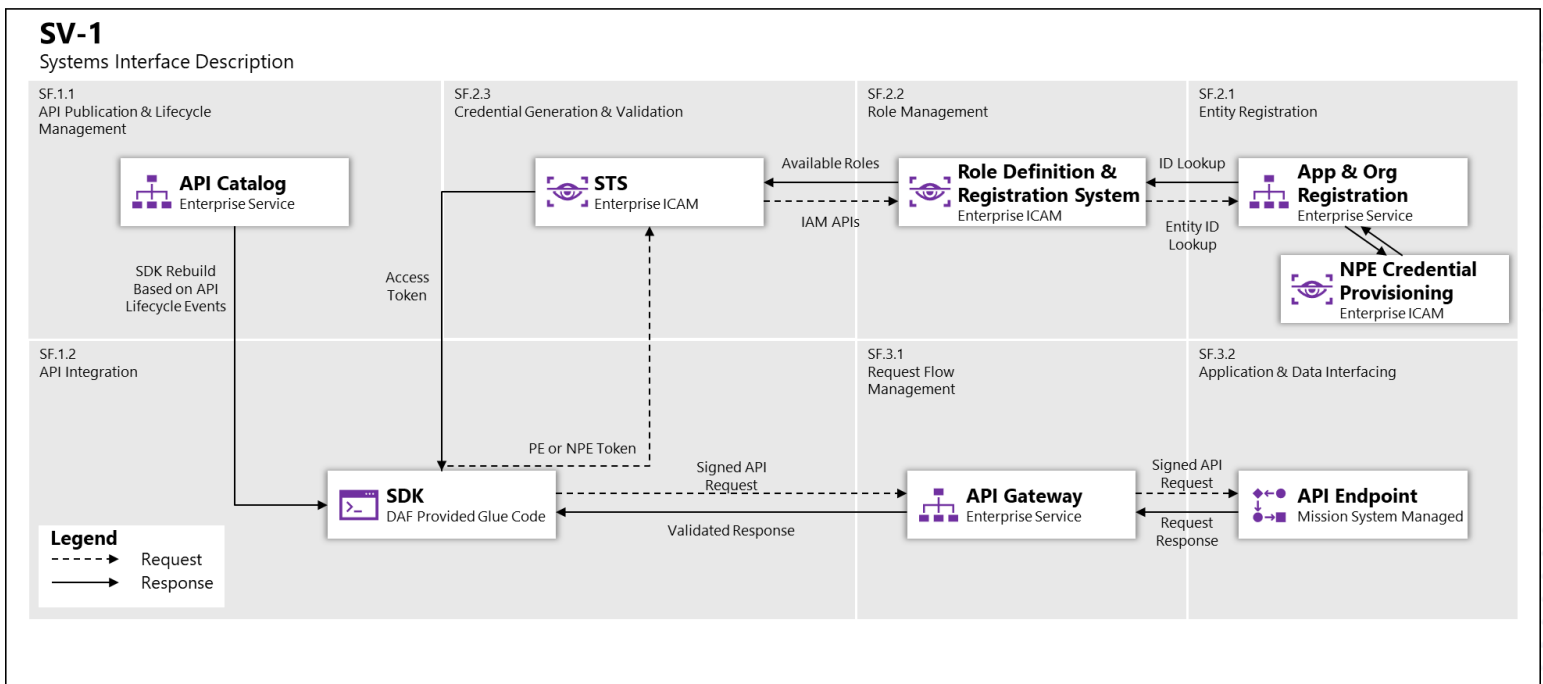


Figure 3: The SV-1 Systems Interface Description depicts the identification of system resource flows and their composition



SV-6

Systems Resource
Flow Matrix

ID	Interfaces		Exchanges			
ID	Requesting System	Receiving System	Returned Data	Type	Protocol	Frequency
INT.1.1	API Catalog	SDK Build System	Registered API Updates	JSON or Smithy Updates	SDK Rebuild & Publication	Lifecycle Events
INT.1.2	Role Definition / Registration System	App & Org Registration	Entity ID Lookup	ICAM TBD	TBD	User Request
INT.1.3	STS	Role Definition / Registration System	Available Roles	IAM API	HTTPS + System Role JWT	User Request
INT.1.4	SDK	STS	Access Token	JWT	HTTPS + PKI	User Request
INT.1.5	SDK	API Gateway	Validated API Response	JSON	HTTPS + JWT	User Request
INT.1.6	API Gateway	API Endpoint	API Response	JSON	HTTPS + JWT	User Request

Table 2: The SV-6 Systems Resource Flow Matrix shows the details of resource flows among systems, the activities performed, and the resources exchanged

3.1. Secure Data Access & Sharing

The three system functions described in Figure 1 – **Interface Management**, **Entity & Access Management**, and **Data Sharing Management** – deliver Secure Data Access & Sharing by standardizing and facilitating real-time system-to-system communication. Each system function is delivered by a suite of specific system interfaces and are described in their own sections below.

3.2. Interface Management

Interface Management describes the services that register, publish, and manage the lifecycle of an API so that consumers can incorporate them into their systems.

3.2.1. API Publication & Lifecycle Management

To expose data for retrieval by APIs, publishers will register their APIs into an enterprise catalog so that consumers know what they can incorporate into their applications. Registration will include information about the endpoint, what data it expects, and the data it returns. The catalog also allows publishers to update their APIs and handles version control and other lifecycle events.

3.2.1.1. API Catalog

The API Catalog is an enterprise service that allows publication and lifecycle management of APIs. To upload to the catalog, publishers will use a facility such as a web user interface (UI) or a Git repository. The catalog includes records of an API's technical documentation, input and output expectations, and associated endpoints. DAF API policy will state that a publisher must maintain backwards compatibility for a set amount of time within a major version or create a new version. When creating a new version, the publisher must deprecate the existing version and allow for transition time. The DAF will first develop an API Catalog prototype based on exposing existing data sets currently housed by major data platforms. Outputs from the catalog will drive the configuration of the API Gateways.



The catalog also acts as the canonical source from which the DAF builds the SDK and any environment-specific add-ons. How the system builds the SDK will be transparent so that developers can test their APIs pre-publication with prototype language bindings and can be assured of the output. The DAF intends to survey available services and commercial products to align to best practices prior to fielding.

3.2.2. API Integration

Software developers are the principal consumer of APIs and associated services. The DAF can facilitate this consumption by providing an SDK available for download to integrate into applications with different versions for all classification levels. Since API Catalog entries are meant to generate language-specific SDK bindings in an automated fashion, the SDK provides enormous convenience to both consumers and publishers.

3.2.2.1. SDK

The SDK exposes a language-specific interface for every supported API registered in the catalog. It facilitates integration of the entire architecture by providing helper functions for interaction with supporting systems such as ICAM. This first-party software supports automated adherence to DAF API Standards & Policy by including facilities that enable clients to stay within rate limits and properly handle errors and appropriate retries. Because the SDK draws information from the API Catalog that receives constant API updates and additions, it will be rebuilt periodically and published to consumers for update.

As a set of programming language bindings that will support major languages, the SDK will be distributed through standard library management systems for each language (PIP, NPM, etc.). To support APIs at different classification levels, the SDK should have extension hooks so that developers can integrate non-public APIs into a publicly distributed SDK. While there are some open-source tool kits that help build an SDK, there may be commercial products that perform more of this work as part of an integrated solution. The DAF will need to establish an outreach office to maintain the SDK publication system and support the development community.

3.3. Entity & Access Management

Entity & Access Management supports the security of software interfacing by describing the rules and methods that govern how a person or system can consume an API. The services that comprise Entity & Access Management are designed to manage the volume of authorization traffic at scale.

3.3.1. Entity Registration

To ensure API requests are secure, both publishers and consumers must have an agreed upon way to authenticate and authorize API traffic. The DAF will provide an enterprise process called Application & Organization (App & Org) Registration to manage the assignment of publishers and consumers into groups that may contain both Person Entities (PEs) as well as Non-Person Entity (NPE) software systems. This will enable groups of people as well as group-owned software to be granted permissions on specific resources and receive credentials. In addition, organization registration is required to publish APIs and access API lifecycle management tools.



3.3.1.1. App & Organization Registration

As a part of the ICAM system, Org Registration allows for definition of the business roles and grouping necessary to manage API permissions. App Registration should require a valid Org registration so that applications are tied to business entities rather than people who may shift roles over time. Both App & Org Registration are enterprise ICAM components that support Role-Based Access Control (RBAC) and have other purposes outside of the API ecosystem. App & Org Registration is necessary to structure the permissions inside of the Role Definition / Registration System. Development will be tightly coordinated with the Enterprise ICAM team and will hinge on requirements on the authoritative data attribute fulfillment inside of ICAM. When Attribute-Based Access Control (ABAC) is mature enough, the system will not require explicit permissions on organizations.

3.3.2. Role Management

Registered consumers must also have the proper set of roles assigned to them for access to specific APIs. Publishers set access permissions for their APIs based on roles in the Role Definition / Registration System. This system allows a publisher to set a wide range of permissions from broad group membership for any registered user all the way to explicit individual approvals for more restrictive APIs. Regardless of whether a consumer is a PE or an NPE, they must have an appropriate entitlement granted through the Role Definition / Registration System to access an API.

3.3.2.1. Role Definition / Registration System

In the Role Definition / Registration System, a publisher chooses the coarse-grained permissions for the API Gateway to enforce on their API, as defined in the API Catalog. Management of these roles should be flexible enough to cover either broad group requests or explicitly approved individual requests for more restrictive APIs. The System will be managed by the ICAM team and have the same global entitlement management elements as the access management system that replaces the DD2875.

3.3.3. Credential Generation & Validation

Like other elements that support a Zero Trust Architecture, the DAF must incorporate strong identity foundations in the API marketplace. As a result, the DAF will support credentialing API consumers inside of the ICAM system. To do so, the API Architecture uses an STS to verify identity from PEs and NPEs and issue a unified but short-lived access Token to make requests.

3.3.3.1 STS

The purpose of the STS is to enable people and software to transform long-lived PKI credentials into lower-risk access Tokens that are tightly scoped to specific API operations. By performing this operation, the shorter-lived Token can be more conveniently consumed by an SDK. The STS is a special class of API that securely returns credential information rather than data or API results. Consequently, it will likely use Mutual Transport Layer Security (TLS) to help prevent man-in-the-middle attacks so that credentials cannot be stolen during transit. The Mutual TLS component of the STS API simplifies the role of the API Gateway to accept a single Token-based credential that can be more easily scaled.



The STS API should mint API Tokens with entitlements that are a subset of an organization's total possible entitlements. This concept of using only the roles that are necessary to accomplish a task further limits the impact of credential theft. API Tokens should also have a considerably shorter lifespan than organization credentials with some flexibility for refresh. However, Tokens are ultimately time-bounded by DAF API policy to limit exposure if credentials are mishandled. It may be advisable to constrain the locations and platforms from which organization or personal credentials may be used to generate API Tokens, for example only allowing from specific IP ranges or from government-managed end user devices.

For particularly complex Cloud architectures that need to consume DAF APIs, the DAF suggests the use of a Token Broker. A Token Broker bridges the use of a hardware-protected PKI credential that represents the overall software system to Cloud resources that can consume an STS-provided Token received as a part of its execution context. Rather than having a system in which everything needs access to the organization's master credential, having a gatekeeper process through a Token Broker is a suggested pattern to properly manage an organization's set of entitlements and provide least privilege when generating Tokens.

The DAF is considering pursuing JSON Web Token (JWT) as the API Token format but is examining industry best practices and software support for the STS. DAF Enterprise ICAM will manage the STS. Token formats should be easy to work with inside of modern programming languages as well as mirror design patterns from major Cloud and software providers who have implemented complete IAM ecosystems.

3.4. Request Management

Request Management enables publishers in the API ecosystem to better support the high volume of requests incurred by participating in a modern, interoperable environment. It does so by automating adherence to DAF API standards and applying standard web scalability techniques to the enterprise architecture.

3.4.1. Request Flow Management

To ensure effective request flow management, the DAF will establish enterprise API Gateways that filter, validate, and route API requests. The gateway ensures requests are valid, going to a known endpoint, have basic authorization, and are within policy. It also validates the response from the endpoint and sends it to the requestor, supplying the appropriate error code if the endpoint is unable to respond to the request. Managing the flow of requests with an API Gateway not only eliminates the challenge of unknown endpoint locations but also adds another layer of request validation to increase the security of data sharing.

3.4.1.1. API Gateway

The API Gateway routes and load-balances requests from consumers to the appropriate endpoint, filtering invalid, unauthorized, or out-of-policy requests to reduce the burden on publishers. Consumers also benefit from the API Gateway as it eliminates the need for them to know where an endpoint lives. The API Gateway receives information about the universe of supported APIs and associated endpoints from the API Catalog and should be updated in near-real time based upon lifecycle events.



The API Gateway first receives a signed request – a JSON Web Token (JWT) – from the consumer, facilitated by the SDK, where it then makes a coarse-grained API access decision. Unlike fine-grained access decisions that will need to be handled by the endpoint, a coarse-grained decision considers only roles granted to the API Token. The shorter-lived JWTs that sign API requests are provided by the STS and are requested by using personal longer-lived Common Access Card (CAC)/Personal Identity Verification (PIV) or Non-Person Entity (NPE) credentials. API Gateways act as a standardized Policy Enforcement Point (PEP) and Policy Decision Point (PDP) for application traffic and validate API standards such as request format, size, and transmission rate. Valid requests are then forwarded to the endpoint for response; however, caching of certain request-response pairs may be desirable. In addition to filtering requests, the gateway will provide a centralized facility for monitoring and analytics for integrations with security operations.

The DAF will provide both deployable gateways as well as enterprise gateways. It will explore existing vendors and look for integrated solutions that include API Catalog functionality. Deployable gateways will support both development environments as well as deployed architectures. API Gateways for integrating the major DAF data platforms into an enterprise API ecosystem may require a specialized adapter inside of those platforms.

3.4.2. Application & Data Interfacing

Mission Systems manage their own API Endpoints, receive requests from API Gateways, and return the appropriate response. When necessary, the API Endpoint is responsible for fine-grained authorization of API requests. The Role Definition / Registration System and ICAM system provide supporting infrastructure for this entitlement management.

3.4.2.1. API Endpoint

The API Endpoint is the mission-provided infrastructure that translates incoming API requests into the specific application logic or data request that each API requires. It is not a specified technology or product but is likely co-located with the application infrastructure for the rest of the system that is providing the API. The DAF will explore best practices for endpoint management and configuration to 1) make it as easy as possible to run an endpoint; and 2) consistently respond to requests to ensure a good developer experience and a resilient, integrated system. DAF API policy will outline expectations for endpoints to conform to such as appropriate return codes as well as best practices for how to ensure a request comes from a valid API Gateway.

If the DAF does not implement an enterprise gateway at the outset of this API journey, early adopter endpoints must be willing to take on those gateway responsibilities such as coarse-grained authorization and request format validation. Consumers that participate in the early API experimentation phases must keep in mind the eventual migration to an enterprise gateway and must carefully coordinate application cutovers. Additional responsibilities on the endpoint, such as the specific processes around fine-grained authorization, are subject to feedback from stakeholders and maturation as best practices emerge.



4. Standards Overview (Std-V2)

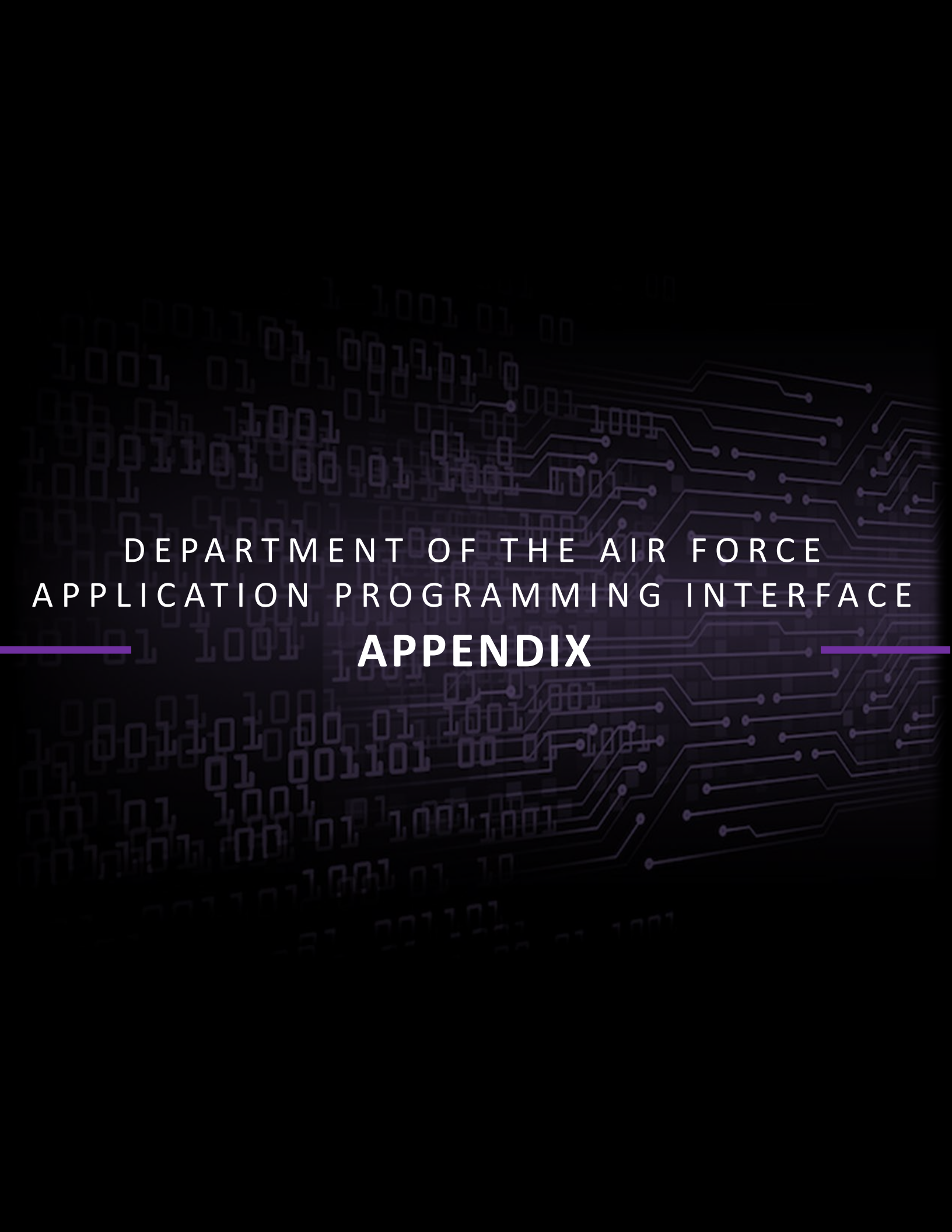
The Standards Overview describes desired standards / policy the DAF will implement as it builds out the API solution.

StdV-2

Standards Forecast

Service	Standards/Policy	Description
API Catalog	Backwards Compatibility	What constitutes a backwards incompatible change
	Lifecycle Phase Definitions	Lifecycle phase definitions to include business rules containing minimum durations for deprecation and sunseting
SDK	Language Specific Standards	Object structure and naming conventions appropriate to each language binding
	SDK Distribution Mechanisms	SDK packaging and release mechanics to include distribution on secure networks
App & Org Registration	Organization Definitions	How to define organizations and how they interoperate with existing organizational constructs
	External Organizations	How to register and sponsor external organizations
	Credential Validity Duration	Set duration for how long organizational credentials remain valid
Role Definition / Registration System	Entitlement Standards	Name spacing standards, lifecycle events, and suggested best practices for least privilege
	NPE Policy	Define lifecycle and business management rules
	RBAC & ABAC Approvals	How approval process works for RBAC & ABAC
STS	Token Standards	Token format and lifecycle standards
	Credential Security	DAF-wide agreement on best practices for PKI and JWT credential security
API Gateway	DAF API Response Standards	Standardized formats and responses
	Request Filtering	Rate limit standards, service default standards, response time standards, how to track environmental characteristics
API Endpoint	Versioned Policy	Describes the API standards version that the Endpoint implements

Table 3: The StdV-2 Standards Forecast presents rules that will constrain activities and their performers in the future

The background of the page features a dark, textured design. It includes faint, glowing binary code (0s and 1s) scattered across the surface, along with intricate, light-colored circuit board traces that meander across the lower half of the image. The overall aesthetic is high-tech and digital.

DEPARTMENT OF THE AIR FORCE APPLICATION PROGRAMMING INTERFACE **APPENDIX**



Acronym	Meaning
API	Application Programming Interface
CAC	Common Access Card
CIO LOEs	Chief Information Officer Lines of Effort
DAF	Department of the Air Force
DDIL	Denied, Degraded, Intermittent, Limited
DNS	Domain Name System
DoD	Department of Defense
DoDIN	DoD Information Network
EIEMA	Enterprise Information Environment Mission Area
IAM	Identity & Access Management
ICAM	Identity Credential & Access Management
IT	Information Technology
JSON	JavaScript Object Notation
JWT	JSON Web Token
Mutual TLS	Mutual Transport Layer Security
NPEs	Non-Person Entities
PDP	Policy Decision Point
PE	Person Entity
PEP	Policy Enforcement Point
PIV	Personal Identity Verification
PKI	Public Key Infrastructure
RBAC / ABAC	Role Based Access Control / Attribute Based Access Control
SAF/CN	Office of the CIO
SAF/CND	Chief Data & AI Office
SDK	Software Development Kit
STS	Secure Token Service
VAULTIS	DoD Framework for Visible, Accessible, Understandable, Linked, Trustworthy, Interoperable, and Secure data
Web UI	Web User Interface
ZT	Zero Trust

Table 4: The Acronyms table defines all major acronyms used throughout this document



Submit your feedback here
<https://www.safcn.af.mil/Contact-Us/>