CSAP

COMMON SECURITY ARCHITECTURE
for PRODUCTION

VERSION 1.2

PART 4:
SECURING SOFTWARE-DEFINED WORKFLOWS

Motion Picture Laboratories, Inc. October 18, 2022

Contents

1 Introduction 1
1.1 Terms and Abbreviations 2
1.2 Icon and Shape Definitions 2
1.3 References 2
1.4 Workflow Diagrams 3
2 Security Goals 4
3 Software-Defined Workflows 6
3.1 Workflow Management 6
3.2 Discussion 7
3.3 Initialize 9
3.4 Execute 10
3.5 Automation 11
4 Authorization Rules 12
4.1 Authorization Rule Lifecycle 13
4.2 Authorization Policy Templates 14
4.3 Global Policies 15
4.4 Authentication 16
4.5 Mutual TLS 17
5 CSAP and Example SDW Components 18
5.1 Securing Services and Infrastructure 18
5.2 Message System 20
5.2.1 With Registration of Producers and Consumers 21
5.2.2 Without Registration of Producers and Consumers 21
5.2.3 Message Security 22
5.3 Asset Management 23
5.4 Identifier Resolution 24
5.5 Asset Retrieval 24
5.6 Protecting The Integrity of Asset Management and Resolution 26
5.6.1 Change The Table Functions 26
5.6.2 Look Up 27

1 Introduction

Part of the MovieLabs 2030 Vision is to remove burdensome, repetitive, and mundane tasks by automating and delegating them to software processes. There are several benefits to this. Most notably it frees up time for storytellers to do what they love – being creative – and cuts out unnecessary complications often found in moving files or metadata, checking transfers occurred correctly, calling other departments for status reports or vendors to check orders. MovieLabs' objective is to enable flexible, dynamic workflows that can be changed and modified whilst making productions dramatically more efficient.

These are some of the most complex problems being solved because they require several pre-requisites. Many of these are explained in the MovieLabs Software-defined Workflows paper[1] laying out some of the needs for Software-Defined Workflows.

The MovieLabs Common Security Architecture for Production, CSAP, is a workflow-driven security architecture for production in the cloud. It is a zero-trust architecture with a deny-by-default security posture and CSAP authorization policies authorize activities. Workflow driven means that security policies are created in response to the immediate requirements of the workflow.

CSAP addresses the security of software-defined workflows (SDW). CSAP secures the SDW, and the workflow management at the core of the SDW initiates the creation of authorization policies that authorize activity.

CSAP is presented in six parts:[2]

Part 1: Architecture Description the main architecture document.

Part 2: Interfaces describes the possible interfaces between the modules in a canonical form.

Part 3: Security Levels presents a metric-based approach to scaling security.

Part 4: Securing Software-Defined Workflow is this document.

Part 5: Implementation Considerations discusses some of the options for implementing this architecture.

Part 6: Policy Description

This document defines how CSAP and SDW work hand in hand. It is assumed that the reader is familiar with the published parts of CSAP, currently 1 to 3, and does not reiterate the concepts described in those parts.

[1] https://movielabs.com/production-technology/software-defined-workflows/

[2] Note: Parts 5 and 6 have not been published as of October 2022.

1.1 Terms and Abbreviations

Authentication is the security mechanism used to validate an entity's identity by a trusted authority. The entity might be a user, a service, a device, an application, etc.

Authorization is the security mechanism used by a trusted authority to determine whether an entity can perform an action.

A *Creative Work* is uniquely identified production.

An *Asset* is a physical or digital object or collection of objects specific to the creation of a Creative Work. *This is the media definition of the word asset, and not the definition used in cybersecurity where the word asset means any data, device, or other component (hardware or software) that supports information-related activities.*

A *Participant* means the entities (people, organizations, and services) that are responsible for the production of the Creative Work.

A *Task* is a piece of work to be done and completed as a step in the production process.

A *Device* is an execution platform, for example, a serverless platform, a server, a virtual machine.

ARDS is the abbreviation of the Authorization Rule Distribution System, a CSAP core security component.

1.2 Icon and Shape Definitions

The shapes and icons used in the diagrams in this document are part of the MovieLabs Visual Language.

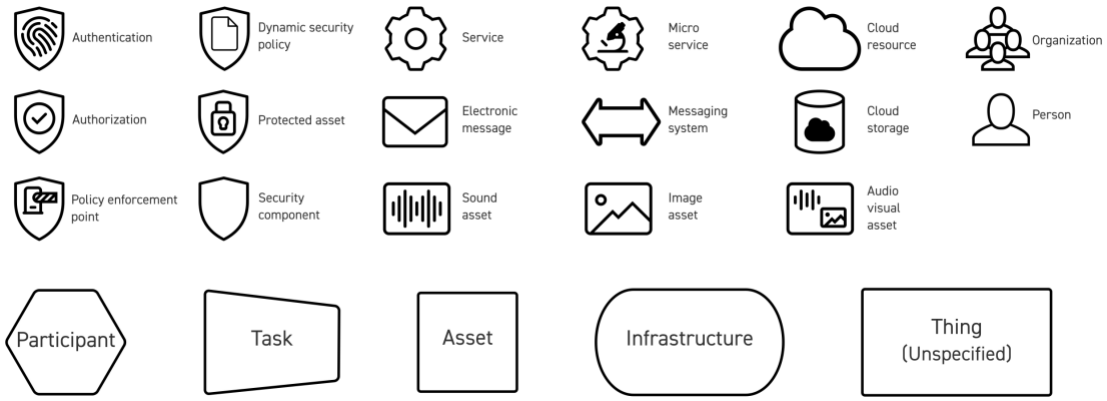


Figure 1-1 Selection of the visual language icons and shapes used in this document

1.3 References

[The Evolution of Production Workflows], MovieLabs, 2020

[Ontology for Media Creation], MovieLabs, 2021-

[Visual Language for Media Creation], MovieLabs, 2021-

1.4 Workflow Diagrams

Workflow diagrams in this document are simplified and they show no more detail than is necessary to explain the point being made about the security using the simplest possible workflow diagrams and descriptions. This document is about CSAP, not the workflows themselves, and mission critical aspects of workflows may be missing. The workflows described in this document are not intended to be comprehensive or complete.

2 Security Goals

Our definition of the purpose of security in media production is twofold.

> **Protect the media production environment or ecosystem.** This is the domain of information security, namely protection from malicious and unauthorized access. The primary goal is the prevention of unauthorized data access and denial of service attacks including ransomware.
>
> **Protect the integrity of the media production workflow**. Protecting workflow integrity means that each workflow activity is carried out by the right participants, on the right device, using the right software and during the appropriate time frame. When we talk about CSAP scalability, we mean that the granularity (e.g., the continuum for "everyone is authorized at any time" to "the Art Director is authorized for one hour") is determined by production management.

In other words, protecting the integrity of a workflow means that the activity complies with the intent of the workflow by ensuring a workflow is conducted as intended, using:

- Approved participants whether human or machine
- Approved/designated applications
- Approved infrastructure

Now, cybersecurity is a multidimensional subject and consequently there is overlap. For example, depending on the contextual meaning of "authorized," requiring a user to be authorized is part of both goals but with different purposes in mind. Protecting the media production environment means that a user must be authenticated and authorized to log into a virtual machine and access files. Protecting the integrity of the media production workflow means, for example, that the user is authorized for a defined period to conduct an activity only on a specified system using a specific application.

We view this as two different purposes for production security because the protection goals are somewhat different. Protecting the integrity of a workflow is not about preventing unauthorized exfiltration of media assets as that is taken care of by protecting the media production environment.

Examples of protecting workflow integrity are:

- Color grading must be done using Baselight and not another grading application such as Resolve. Here workflow integrity means ensuring the correct application is used
- Editing must be done using Avid Media Composer version 2021.3 and not 2021.6 because 2021.6 changed the default behavior for changing workspaces. Here workflow integrity means ensuring the correct version of the application is used[3]
- The editor contracted by the production is the person that is editing the content

Some requirements might contain elements of security and integrity. For example, the editor has finished on a piece of content, and it is ready to be reviewed. If an authorization rule is required to start

[3] It is to be noted that application plug-ins present their own set of challenges which are outside of the scope of this document.

the review activity, including authorization to access to the content to be reviewed, the rule could be enabled only when the editor has finished. This would prevent premature review of the content.

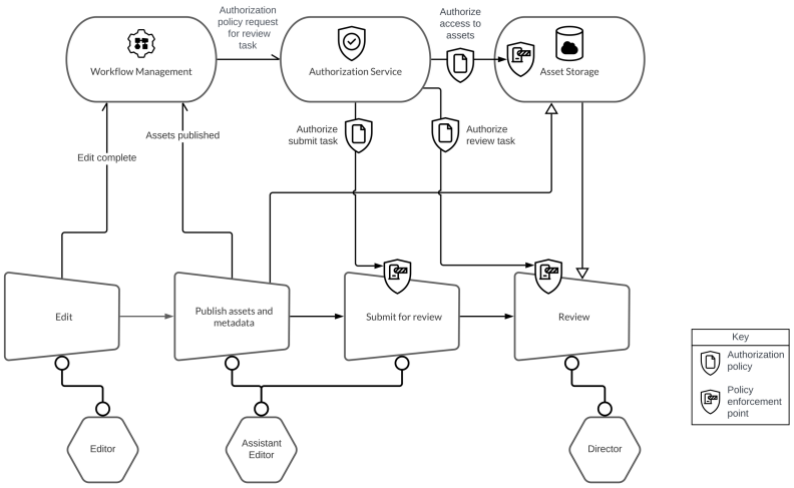


Figure 2-1 Example workflow

In this example, workflow management is notified when the edit task is complete. When the assets have been published which, in this case, means moving them to the asset storage, workflow management requests authorization for the next steps.

1. The task *submit for review* is authorized. This protects the integrity of the workflow, the *submit for review* task cannot start until the workflow management requests it be authorized which will not happen until the *publish assets and metadata* task has completed
2. Access to the asset storage by the *review* task, is authorized. This protects the assets

3 Software-Defined Workflows

The future of production will rely on highly configurable workflows that can be continually adapted to support new creative needs of the production, implement new business requirements, or interact with new partnerships. Production teams will design and directly manipulate workflows, and software will manage the processes of collaboration and orchestration.

Anyone designing a workflow will have the ability to choose which tasks are used to perform specific functions, what assets and associated information those tasks communicate, which participants are involved, and what the rules are to move or gate the process. Examples of rules that can be built into workflow automation include "Raw image captured, invokes proxy encoding service" and "director's approval required at this point." We use the term software-defined workflows (SDW) to broadly describe workflows that fit this model. For discussion, we will use the following definition:

> A *software-defined workflow* uses a highly configurable set of tools and processes to support creative tasks by connecting them through software-mediated collaboration and automation.

Software-defined workflows make it practical to develop reusable components and to automate aspects of the workflow that are currently manual.

Since workflow drives security, CSAP is there to permit authorized activity and no other; what is authorized comes from workflow management. We address this further in Section 4.

3.1 Workflow Management

In any part of an overall workflow there is almost always something or someone that is scheduling work, creating instructions/work orders, and tracking that work. Increasingly, more of these functions are supported by software connected to automated services. We generally call them workflow management. There will rarely be a single workflow manager that manages all aspects of a workflow and, most commonly, each portion of an overall workflow will have its own. In a world of workflows that can be decomposed into smaller workflows, each workflow manager will need to coordinate with the workflows that precede, succeed or operate in parallel.

This coordination usually happens between them but can also be managed by another layer of workflow management. In Figure 3-2 we have an example where one workflow management entity is managing other workflow management entities that each run a discrete workflow which means that something can track the state at various levels and communicate it upwards and downwards.

There is an execution layer below workflow managers that performs storage and compute orchestration, often under their direction or acting in response to requests.

This document is concerned with securing software-defined workflows, and as such we are concerned with the use of the workflow management tools.

We use the term **workflow management** to mean the whole process of managing a workflow including procedures such as hiring crew, contracting with vendors, etc., and **workflow manager** to mean an entity that initializes and manages one or more workflows.

3.2 Discussion

In this document we will use a dailies workflow as a reference.

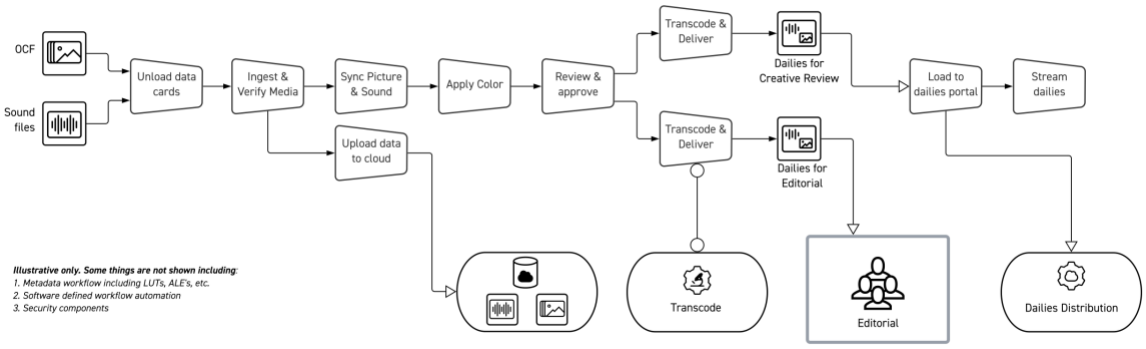


Figure 3-1 Reference dailies workflow

This example workflow has three component workflows and one way of managing the workflow is using workflow management for each of the component workflows.

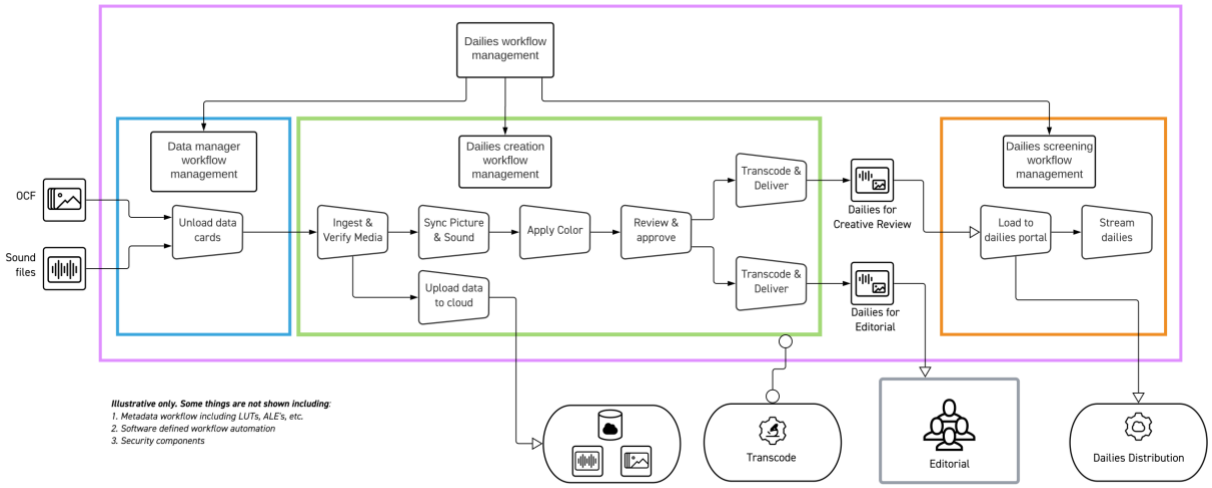


Figure 3-2 Example of workflow management

And of course, the dailies workflow is a part of the whole production workflow and works in conjunction with the other workflows.

Before we can look at how CSAP interacts with the workflow management, we need to describe the workflow itself. Let us focus on the dailies creation workflow, the green box in Figure 3-2, which may be referred to as the dailies department.

1. The production sets up the department
    a. The crew are hired
    b. Office space is arranged
    c. The equipment is procured, delivered to the offices where the dailies department will be located, and configured
    d. The security level is determined for each part of the workflow.
    e. System accounts are created for the crew
2. The crew agrees on their workflow and how they will use resources, for example by drawing a data flow map. This is an intra-departmental, single domain, activity.
3. The crew and the data management department agree on data flow, for example, a watch folder. Again, this may be a data flow map. This is an inter-departmental, multi-domain, activity.
4. The workflow waits
5. When camera files (OCF) and sound files arrive from the set, the workflow starts
    a. Data ingest and verification starts
    b. When data is verified, sound sync starts, and OCF and sound files are uploaded to the cloud
    c. When sound is synchronized complete, color grade starts
    d. When color is complete, the director reviews and approves
6. Dailies are transcoded for editorial and creative review, and delivered to editorial and dailies distribution respectively

The workflow iterates until the production wraps

We can group these steps:

- 1-3 are about set-up and are largely one-off. They are about the workflow management initialization, meaning getting things ready for work to commence. As we go down the hierarchy, the set up gets more specific but, today, probably less formal.
    - We will call this **initialize**
- 4-6 are event driven
    - Transition from step 4 to step 5a is triggered by the arrival of data from the set
    - Steps 5b to 5d are triggered by the completion of the step before
    - We will call this **execute**

Going back to Figure 3-2 and adding detail we have this:

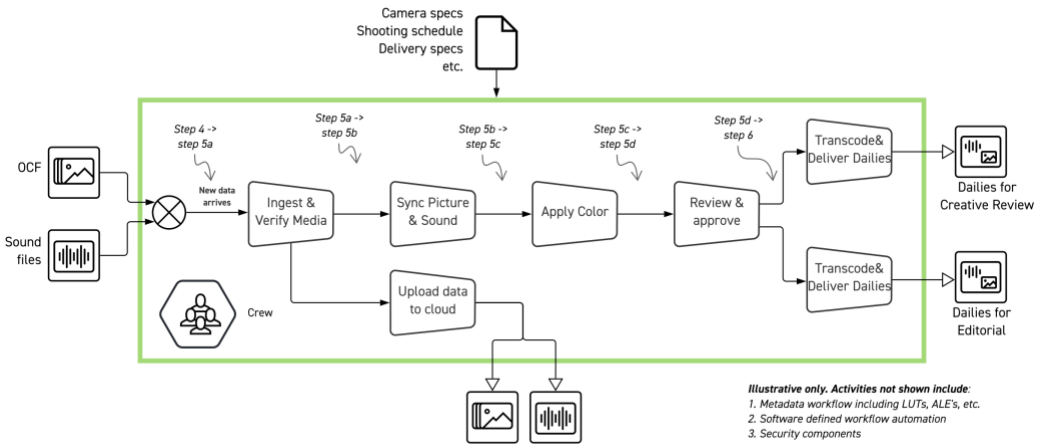


Figure 3-3 Dailies dept. workflow

3.3 Initialize

In this example, which is not atypical, initialization in the form of camera specifications, shooting schedule, crew selection, delivery specifications, etc., is managed by the production. Once the execution phase starts, production management will update the department with shooting schedules, location moves, etc., but those are extensions of the initialization. Initialization itself has workflow management, although it might be distributed and not have formal boundaries. In many cases, that is the scope of the workflow manager involvement. Although the workflow will still be monitored, that is more likely an assessment that the workflow is producing the correct output in a timely manner.

It is during initialization that the CSAP security level is chosen for each part of the workflow (there is no requirement that the same level is used throughout). Selection of security level is based on risk assessment and risk tolerance.

Adding security into our previous description of the initialize phase:

Step	Workflow	Security
1	The production sets up the department	Identity management accounts are created for each crew member. Security levels are determined. Global security policies are defined/acquired. CSAP interoperable systems and services are procured
2	The crew agrees on their workflow	Authorization policy templates are created. Positioning and provisioning of PEPs is determined

Step	Workflow	Security
3	The crew and the data management department agree on data flow	Authorization policy templates are created Positioning and provisioning of PEPs is determined

Initialization has both manual and automated aspects. Auto-scaling of compute resources is, as the name implies, automated. Some tasks, such as assigning artists, could be either manual or automated using a scheduling system. And the first time a workflow is established, it is likely to be a manual process.

3.4 Execute

Execution is often largely event driven and, once the department agrees on its workflow, adjustments are made to accommodate new requirements from the production management or to improve the workflow.

Adding security at the most granular level of authorization rules into our previous description of the execute phase give us:

Step	Workflow	Security
4	The workflow waits	
5	Workflow starts when camera and sound files arrives	
5a	Ingest & verify media starts	An authorization rule authorizes data ingest
5b	When data is verified, sound sync starts, and OCF and sound files are uploaded to the cloud	An authorization rule authorizes the activities sync sound and upload to cloud
5c	When sound is synchronized complete, color grade starts	An authorization rule authorizes the activity apply color
5d	When color is complete, the director reviews and approves	An authorization rule authorizes the activity director review
6	Dailies are transcoded and delivered to editorial and for creative review	On director approval, an authorization rule authorizes transcode and delivery

Note: This is illustrative, and it is unlikely that authorization rules would be used at every step.

A workflow manager would be needed if, for example, resource scheduling was being done outside of the workflow. If the dailies department was servicing more than one production then crew, work and shared infrastructure would need to be scheduled.

3.5 Automation

While software-defined workflows are automated to at least some extent, there is no CSAP requirement for automation in the workflow. CSAP Part 3: Security Levels discusses the need for automation between CSAP security components.

4 Authorization Rules

CSAP is a deny-as-default zero-trust architecture that has two basic rules:

1. Nothing can be part of any workflow unless it has been authenticated
2. Nothing can be part of a particular workflow unless it has been authorized

Consequently, everything must be authenticated, and all actions must be authorized.

Unlike conventional security policies which are relatively static, CSAP *authorization rules* enable the principle of least privilege with a temporal component and authorization should always be for the minimum period to complete a task.

Let us look at the big picture relationship between the CSAP architecture and the software-defined workflow. Here we have broken down the functions of the workflow management into a view of the components that aligns with the CSAP services

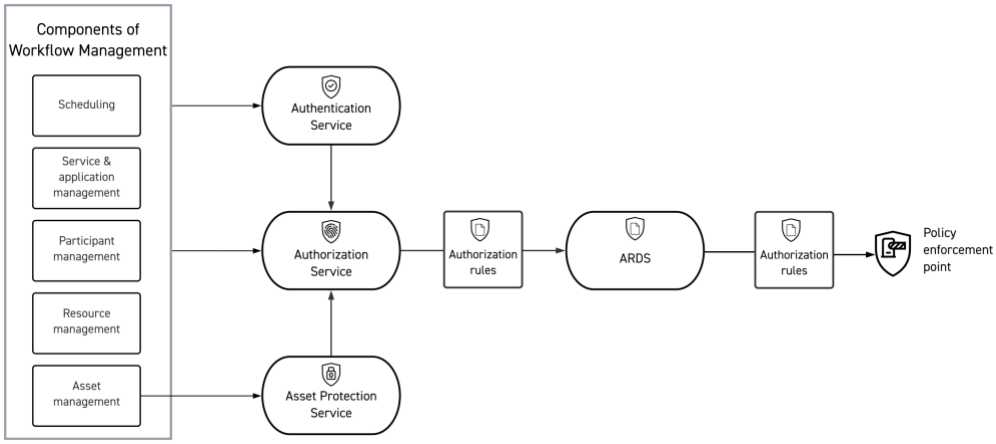


Figure 4-1 CSAP Connecting points between SDW management and CSAP

In the preceding diagram, the components of workflow management are a way of looking at the role of workflow management and are not meant to imply anything about its design or implementation.

Authorization is conveyed through authorization rules created in the CSAP *authorization service* by applying preconfigured authorization policy templates to authorization requests from SDW workflow management.

CSAP *authorization rules* are constructed around components of a workflow activity which may include:

- Participant
- Device
- Application. For example, software with a user interface, software with API
- Action. For example, edit, start or stop service
- Timeframe. A period delimited by time or event
- Asset

As a rule of thumb, *participant* and *action* are always present, and *asset* is always present if assets are used. Security best practices say that fields should be as specific as possible. For example, the participant should not be "everyone." [4] The *device* and *application* might be combined as, for example, a service or virtual machine configured to run one application.

4.1 Authorization Rule Lifecycle

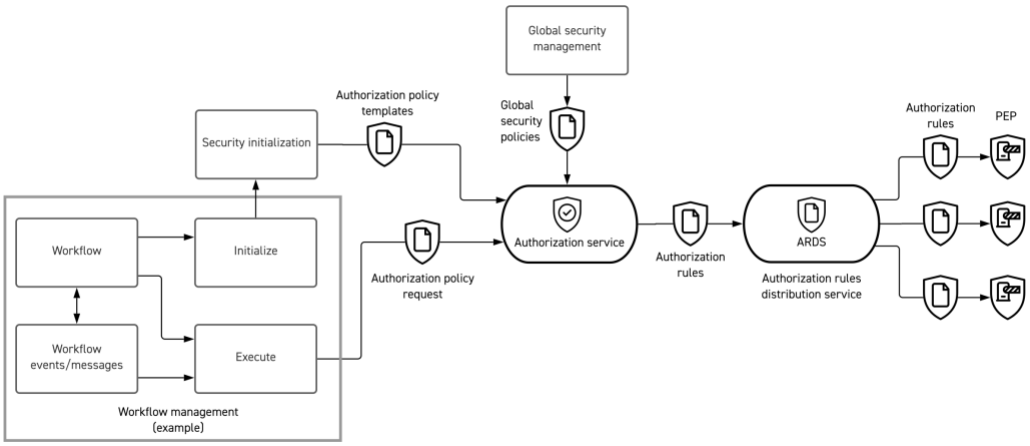


Figure 4-2 Authorization rule lifecycle

Authorization rule creation comes as the result of requests by workflow management, possibly triggered by events and messages that cause the workflow to progress or are created by the workflow, and by automated and manual initialization.

In the figure above we see workflow management driving the process. This diagram is a great simplification, omitting many other initialization tasks such as provisioning identity management, and we do not show any workflow management.

Authorization rules are created by the authorization service by applying authorization policy requests to authorization policy templates. These templates are created, among other places, during the initialization process and when changes are made to a workflow.

The difference between a policy and rule is that a policy is a statement defining what is authorized or what must be denied, and a rule describes a policy in a way specific to the policy enforcement point it is directed to. A policy template is the means to convert from a policy to a rule.

The authorization service ensures that the authorization rules will comply with global security policies. It then sends the authorization rule to the ARDS which distributes it to the appropriate PEPs.

[4] CSAP does not prevent the participant from being "everyone" in the requested authorization, but authorization policy templates and global security policies may not permit that.

4.2 Authorization Policy Templates

The role of an authorization policy template is to facilitate translation by the authorization service from authorization requests received from workflow management into authorization rules that will be distributed to Policy Enforcement Points. One way of looking at authorization policy templates is as a partially "filled in" authorization rule with blank fields that are set by the parameters in the authorization policy request.

An authorization policy template is created by security initialization as a workflow is set-up or initialized.

In the example below, the template is used to create authorization rules for CG artists working on part of a workflow.

Field	Template value	Meaning
Participant	<CG Artist>	A variable, the identifier of an authenticatable CG artist
Device	Windows, AWS	Windows virtual machine running on AWS
Action	Modify	As it says
Application	Maya	As it says
Timeframe	<Task duration>	A variable, the duration of the task
Asset	<URL list>	A variable, an enumerated list of asset URLs[5]

Just to be clear, this is an example showing the use of URLs to locate assets but there are other ways that this can be done.

A request from a workflow manager might look something like this; notice that it only needs to populate the variables in the template:

Field	Request value
Participant	AEFGG678EXAMPLE
Timeframe	4 days
Asset	https://Foobar.local/asset1 https://Foobar.local/asset2 https://Foobar.local/asset3 https://Foobar.local/asset4 https://Foobar.local/asset5

And this would result in an authorization rule that looked like this:

Field	Resultant authorization rule value
Participant	AEFGG678EXAMPLE
Device	Windows, Azure
Action	Modify
Application	Maya

[5] An enumerated list of URLs may not be the optimal way of specifying the assets and is used for simplicity.

Field	Resultant authorization rule value
Timeframe	4 days
Asset	https://Foobar.local/asset1 https://Foobar.local/asset2 https://Foobar.local/asset3 https://Foobar.local/asset4 https://Foobar.local/asset5

In this example, the authorization service has the simple job of substituting variables in the template. A more sophisticated authorization service implementation might have constraints set on allowed values for variables. For example:

- The duration must be ≤ 6 days
- The device can be Windows running on AWS or Windows running on Azure
- The application must be one of a set of authorized versions

This more sophisticated approach protects the integrity of the workflow while giving the workflow manager more flexibility within the specified constraints such as the version of the application.

4.3 Global Policies

The authorization service must ensure that authorization rules comply with global security policies, for example those of the studio that owns the production, and the current security stance (status).

Authorization policy requests may be disallowed when the authorization rule that would be created from the workflow's authorization policy request conflicts with global security policies. If the global security policies result in any element of an authorization policy request being denied, an authorization rule is not created.

For example, the authorization rule for a VFX artist to perform a task that would have been generated at the request of a workflow manager conflicts with a global security policy.

Field	Authorization policy request value	Global security policy
Participant	7EFGG000EXAMPLE	
Device	Personal workstation	Use of personal workstations is not permitted
Action	Modify	
Application	Maya	
Timeframe	4 days	
Asset	https://Foobar.local/asset1 https://Foobar.local/asset2 https://Foobar.local/asset3 https://Foobar.local/asset4 https://Foobar.local/asset5	

In this case, the authorization service cannot create the authorization rule requested using available authorization templates. How the error is conveyed to the workflow manager is an implementation consideration.

In another example, a suspected security breach in the virtual machines used by editorial has been identified by the production's InfoSec group which creates a new global security policy.

Field	Authorization policy request value	Global security policy
Participant	BEFEDITOREXAMPLE	
Device	Virtual machine running on company private cloud	Use of virtual machines on company private cloud not permitted
Action	Modify	
Application	Media composer	
Timeframe	90 days	
Asset	https://Foobar.local/proxy	

Please note that neither of these examples are issues introduced by software-defined workflows or by CSAP. Both issues can happen with today's security models however CSAP has the option of detecting the conflict early on and, subject to implementation, facilitate speedy resolution.

4.4 Authentication

Authentication is a critical part of any zero-trust architecture and, in CSAP, extends to the everything engaged in a workflow. The purpose of authentication is to ensure that something claiming to be an entity that is trusted is indeed that entity. This requires a trusted system, a root of trust, to perform or confirm authentication.

Entity	Trusted system
Users and groups of users (including organizations)	Identity management system such as Azure Active Directory, Okta, and Centrify.
Devices, virtual devices and SaaS[6] services	Certificates in conjunction with a public or private certificate authority
Applications and other software	Software signatures with a verification service.

Certificates used to authenticate services and devices require a certificate authority (CA) to create and manage certificates. The CA might be either:

- A public certificate authority, generally trusted by major browsers and operating systems.
- A private certificate authority, a CA that is trusted by the enterprise.

A private CA may be a service provided by a cloud provider, by a commercial CA provider, or implemented within the organization.

[6] We include both commercial SaaS services and services configured to operate in the same way as a SaaS service.

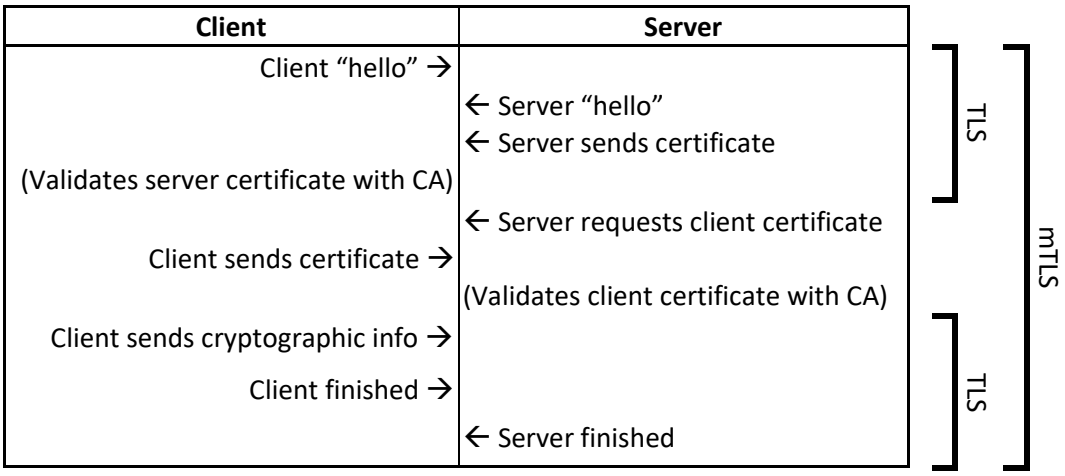
4.5 Mutual TLS

CSAP requires mutual authentication. In Part 5 of the CSAP documentation we discuss the use of mutual TLS [mTLS][7] instead of TLS in more depth but we touch on it briefly here[8].

When a TLS connection is set up between a client and a service, the service presents its certificate to the client and the client can (should!) confirm the certificate is valid with the certificate authority. However, nothing in this process authenticates the client to the service. If that is necessary, for example a bank website needs to authenticate its customers before giving them access to their accounts, a separate unrelated mechanism (e.g., a log in) is used to authenticate the client.

mTLS adds another step to the TLS connection set up where the client presents its certificate to the service which confirms the certificate is valid with the certificate authority. Once an mTLS connection is established, the authentication is mutual and further authentication of the client may not be necessary.

The protocol operates this way:



Communications can now be encrypted.

Checking the validity of the certificates is not part of the protocol but is required for good security.

Importantly, when the client is a user and their device, CSAP requires mutual authentication between the user's device and the service as well as between the user and the server. User authentication is not sufficient because it does not ensure that the user's device is one that has been identified as trustworthy. This is not a new concept, for example, many enterprises only allow corporate issued computers to access email servers. Before the computer can connect to the email server it is authenticated as being a corporate device.

[7] https://en.wikipedia.org/wiki/Mutual_authentication

[8] Mutual authentication is a CSAP requirements; the use mTLS is not a requirement, it is an implementation choice.

5 CSAP and Example SDW Components

In this section we discuss some examples of specific systems that might be part of a software-defined workflow can be secured.

5.1 Securing Services and Infrastructure

An SDW uses a set of services and infrastructure to do useful work.

Role	Components/Services	Infrastructure
Production/workflow management	Scheduling tool	Cloud, data center server, workstation
	Workflow manager	
	Service and application management	
	Participant management	
	Resource management	
	Orchestration services	
SDW platform	Messaging system	Cloud
	Resolver	
	Asset management	
CSAP security system	Authentication service	Cloud
	Authorization service	
	Asset protection service	
	Authorization rule distribution service (ARDS)	
	CSAP supporting components	
	Policy enforcement points	Location/system specific

It is important to understand that a zero-trust architecture is a philosophy and a strategy, not a security product. A zero-trust architecture is implemented using a set of tools and services many of which are used in other security models. However, a zero-trust architecture requires that anything that is authenticated can be trusted[9]. A pre-requisite to trusting a service, system, device, etc., is that is secured and does not rely on security measures outside of the trusted entity, such as a security perimeter[10] around the network the entity is attached to (see Figure 5-1). Securing APIs is discussed in CSAP Part 5: Implementation Considerations.

[9] Anything that cannot be trusted must not be authenticated!

[10] A host connected to a virtual network defined by a software-defined perimeter (SDP) is not relying on the network for security. The SDP controller only permits authenticated and authorized hosts to join the SDP defined virtual network and all traffic between hosts is carried out over mTLS connections. Hosts in the SDP will not respond to traffic originating from outside the SDP and no exchange between hosts in the SDP will happen without the initial mutual authentication of mTLS. The underlying network is not assumed to be secure.

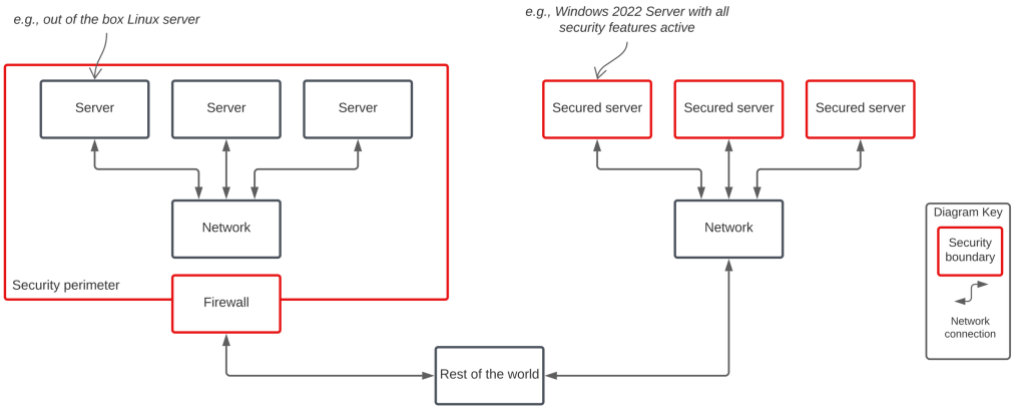


Figure 5-1 Unsecured vs. secured servers

This is reflected in the two sides of Figure 5-1. The three servers in the left side of the diagram may be as secure as the three servers on the right of the diagram but the difference is that on the left of the diagram, security comes from the security perimeter created by the firewall. The three servers on the right of the diagram rely on the security functions installed in them. Thus, the left side of the diagram represents a method of securing the servers that is not suitable for CSAP (or any zero-trust architecture) and the right side represents the security method required by CSAP.

The requirements for controlling access to SaaS services is straightforward although the APIs are only as secure as the SaaS can meet these requirements:

1. Access is only granted when accessed by authenticated entity that is authorized to do so.
2. Access to assets is controlled at an appropriate granularity.

Depending on how the SaaS service operates, those two requirements can be met by the SaaS service itself or by a PEP attached to the service.

When a SaaS service has its own identity and access management (IAM) control, that system should support the setting of authentication and authorization parameters from CSAP services, for example:

- Direct method: SaaS service can read and act on authorization rules
- PEP method: SaaS service authentication and access controls are set through an API by a policy enforcement point.
- Indirect method: The SaaS service uses an external IAM[11] system such as Active Directory, and the authorization rule is used to set authentication and access controls in the IAM (there is no direct interaction between any CSAP component and the SaaS service).

[11] Identity and Access Management

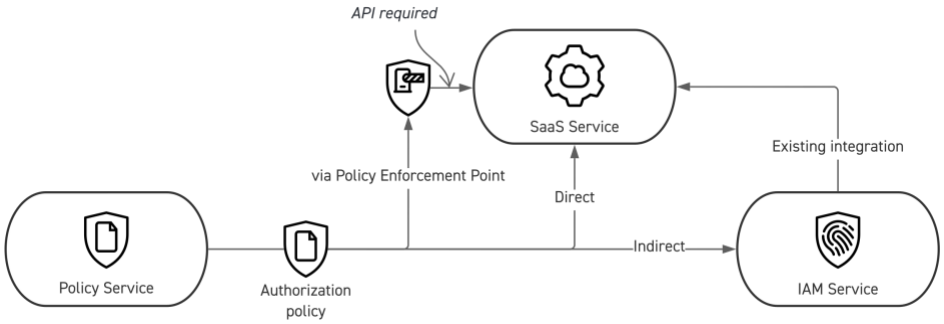


Figure 5-2 Three methods for controlling SaaS security

If no option is available, the solution requires a policy enforcement point acting as a proxy for the service that does not use any of the SaaS service's authentication and access control functions.

The role of the PEP is to ensure that nothing can access a unless it has been authenticated and authorized. That would apply to any participant accessing the service including another service.

5.2 Message System

Message systems are often used in SDW systems for event communication. The function of the message system is, not surprisingly, to pass messages between the constituent parts of a workflow. A message system can be viewed as a set of *named routers,* a generic phrase we use in this document to separate out this description from any particular implementation. We only need to look at the characteristics of a message system and not how it is implemented.

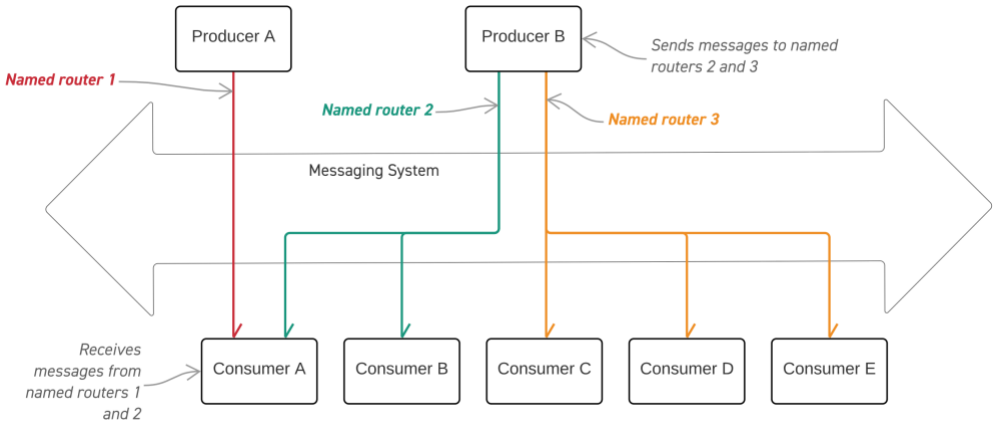


Figure 5-3 Reference messaging system

CSAP requires mutual authentication which means a method, such as mTLS, is used for all communications between the messaging system and producers and consumers.

If the messaging system cannot be trusted to maintain the confidentiality and integrity of the messages it handles then additional measures, for example end-to-end encryption, are needed to protect the use of the messaging system. This is discussed in CSAP Part 5 Implementation Considerations.[12]

5.2.1 With Registration of Producers and Consumers

If the message system controls which producers can send to a named router and which consumers can receive message from a named router, it likely will have some mechanism of registration and subscription.

- A producer registers with a named router to be able to send messages to it
- A consumer subscribes to a named router and receives all messages sent to it

If this is the case, a policy enforcement point associated with the message system will only permit producers to register to a named router and consumers to subscribe to a named router when they are authorized to do so.

If the messaging system is trusted, there is no need for any security action when messages are received by the named router and delivered to consumers. A trusted messaging system is one that can be trusted to:

1. Not allow producers to register or consumers to subscribe to any named router except under the supervision of the policy enforcement point (this means there are no APIs that do not have a PEP)
2. Only accept messages from producers registered with that named router
3. Only deliver messages to consumers of a named router

5.2.2 Without Registration of Producers and Consumers

If registration of producers and consumers is not required, producers can send messages to any named router they can discover, and consumers can receive messages from any named router they can discover.

In this case, the policy enforcement point attached to a named router must only allow messages to be sent to the named router by authorized producers, and only allow messages from the named router to be delivered to authorized consumers. This can be achieved in several ways including:

- On a connection basis where only producers and consumers authorized to use a named router can connect to it
- On a message basis where only allow producers and consumers authorized to use a named router can send messages to or receive messages from it

To make this work, certain things are necessary. For example:

- If control is on a connection basis, the policy enforcement point must be able to determine which named router is being connected

[12] Part 5 has not been published as of October 2022.

- If control is on a message basis, the policy enforcement point must be able to determine which named router a message is going to or coming from

5.2.3 Message Security

A fully secure messaging system will provide the following functions:

- Message confidentiality
  - Prevention of unauthorized access to the contents of the message
- Message integrity
  - Assurance that the message has not been tampered with
- Prevention of impersonation
  - The injection of messages that appear to come from a legitimate producer
- Prevention of message repudiation
  - A producer denying that a message was sent
- Prevention of replay attacks
  - A captured messages is re-sent such that recipients interpret it as a new message

Message confidentiality is a data security issue whereas the mechanisms to prevent impersonation, repudiation and replay attacks protect the integrity of the workflow.

A typical messaging system will use TLS/mTLS connections between the messaging system and its producers and consumers. This provides confidentiality and integrity protection from outside interference and eavesdropping. However, if the message system cannot be trusted with plaintext messages, another security mechanism would be needed to provide end-to-end encryption.

For the rest of this section, we will assume that the message service can be trusted with plaintext messages and that we are protecting messages from external actors.

Whether a messaging system should have all the five characteristics listed above is an implementation decision.

5.2.3.1 Message confidentiality

The use of TLS or mTLS ensures message confidentiality in the transmission of messages between producers and consumers and the messaging system. It does not provide end-to-end message confidentiality.

5.2.3.2 Message integrity and authenticity

TLS or mTLS protect the integrity of messages and prevent impersonation (assuming the messaging system meets our requirements to be trusted) but there may be reason to use end-to-end integrity protection.

Message integrity (detecting whether a message has been altered) and authenticity (determining that the message came from the producer it purports to come from) are achieved by cryptographically signing the message using the producer's private key. The receiving party checks the signature using the

producer's public key. If the message is altered, the signature will not be valid, and the message should be rejected.

This signature is not the same as the use of a mathematical algorithm, such as a CRC code, to detect errors introduced in the transmission of a message. These provide protection from message corruption.

5.2.3.3 Non-repudiation

Repudiation is when a producer denies sending a message.

The mechanisms for message integrity and confidentiality only partially solve the repudiation problem. The producer cannot deny sending the message, but the producer can deny sending the message at a particular time.

Repudiation can be prevented using a time stamp that is cryptographically bound to the message. That means that the time stamp cannot be changed after the fact by the producer.

5.2.3.4 Replay prevention

A replay attack is an attack where the attacker captures a message, re-sends it to the messaging system where it is interpreted by those receiving the messages as a new message. This is partially prevented if messages are only accepted from authorized producers but a misbehaving entity that is registered as a producer and as a consumer can conduct a replay attack.

Replay attacks can be prevented using a session ID or token and a component number. The two mechanisms are not interdependent, and the lack of interdependency means there are fewer vulnerabilities.

For more information on replay attacks see *Malladi, Sreekanth. "On Preventing Replay Attacks on Security Protocols" (PDF). oai.dtic.mil.*

5.3 Asset Management

Asset management in SDWs maintains both metadata about assets and either the assets themselves or their locations. It's useful to separate out the metadata management from the storage and location management functions logically, even when they are implemented together. The metadata management often includes a service that maintains a database of metadata associated with each asset and one or more asset identifiers. Separately, the identifier may be used to look up the location or retrieve the assets. In this model, the lookup of metadata and the lookup of locations are separate logical functions.

The goal in securing asset management is the control of creation, reading, modification and deletion of entries in the asset management database.

While CSAP is also responsible for asset protection, that responsibility is met either at the point where assets are stored (typically using Access Control Lists), or where they are consumed and created (using, for example, asset encryption). Asset encryption supports the case where it may be appropriate for a user (e.g., a production coordinator) to know that assets exist without having access to their essence/contents.

Although asset management does not participate directly in asset protection, asset management's responses to a query may be used to inform the asset protection service which assets are being used and where they are.

5.4 Identifier Resolution

A resolver is needed when the asset management uses asset identifiers and does not hold the location of each asset. Assets are identified by an identifier and a scope within which that identifier is valid, often in the form of a URI. When it is necessary to access the asset,[13] the URI must be resolved to a URL which is used by the application to access to the data.

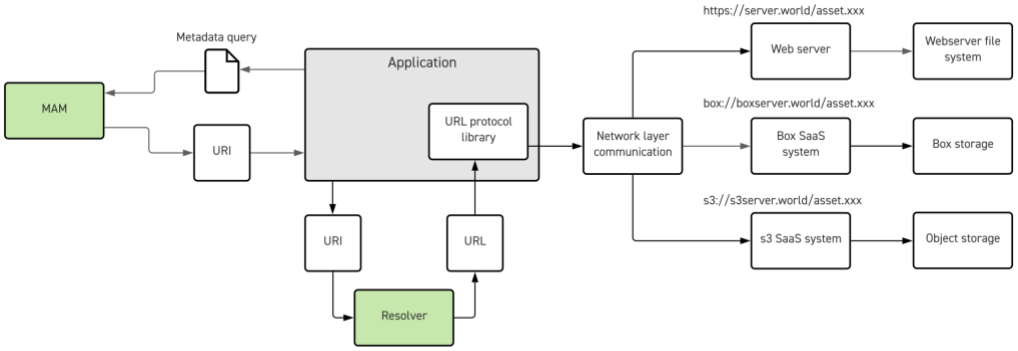


Figure 5-4 Application, asset manager and resolver

The resolver is the system component responsible for resolving the URI to one or more URLs.

The security goal in securing resolution is the control of the creation, reading, modification and deletion of entries in the resolver's database.

5.5 Asset Retrieval

In Figure 5-4, the application has obtained an asset URI from the asset management. To access the asset, it must resolve the URI into the location of the asset and know which protocol to use to access the asset. The application sends the URI to the resolver and the resolver replies with one or more URLs (assuming it knows about the asset). The set of URLs might refer to multiple locations where the asset is stored (e.g., edge caching), to multiple protocols available to access the asset or some combination of the two. For example, two different protocols can be used to access the asset in the same location, or two different locations can be accessed by the same protocol.

However, it may be the case that an application is only authorized to access the asset using a subset of the available URLs. The limited authorization may be for security reasons or workflow reasons[14].

[13] The application may only have an identifier/scope.

[14] Preventing an application getting a URL from the resolver that it cannot use is an implementation topic. There is a security factor where the application is not authorized to access the asset at that URL, and there is an infrastructure factor where the application may not have a network route to the asset at that URL.

For example, the asset exists on multiple clouds and the application is only authorized to access it on one of them. A reason for that might be that the application is only authenticated and authorized on one cloud; another reason might be that the workflow dictates that a particular location must be used to avoid disrupting downstream activities if the asset created by the application is stored on the wrong cloud.

Of course, if the application tried to access a location where it was not authorized to do so, it would fail but how that is avoided or how that event is recovered from is a matter for workflow management.

In this example, we follow assets from the camera to editorial.

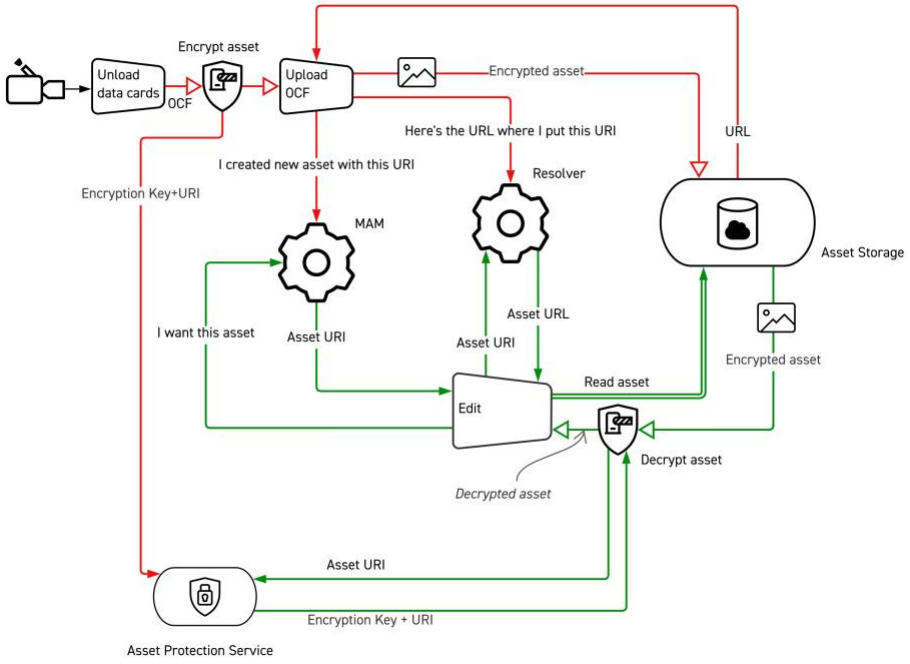


Figure 5-5 Assets resolution life cycle

In the figure, the red lines represent the creation of the asset and the addition of information to the asset management and resolver, and the green lines represent the edit task using that asset.

The creation steps are:

1. The OCF file is unloaded from the camera data cards, encrypted and a URI created.
2. It is uploaded to the cloud asset storage and the URL to access it is determined/created.
3. The asset management is informed for the asset's creation and its URI.
4. The resolver is informed of the URL for the URI.

The use steps are:

1. The edit task queries the asset management and the asset's URI is returned.
2. The edit task queries the resolver using the URI and URL(s) for the URI are returned.
3. The edit task sends a read request for the asset to the asset storage.

4. The encrypted asset is sent to the policy enforcement point associated with the edit task.
5. The policy enforcement point obtains the encryption key from the asset management service and decrypts the asset, making it available to the edit task.

The expectation is that the policy enforcement point would already have the encryption keys cached. The keys are tied to the URI, the URL or some other form of identifier, and all instances of the asset are encrypted with the same key, so the keys can be retrieved from the asset protection service as soon as the URI is known. If an asset has more than one URI, to be precise if an instantiation of an asset has more than one URI, the asset protection must be notified when additional URI are assigned so that it can maintain its mapping between the URI and the encryption key. The asset protection is independent of the location of the asset.

If the asset was not protected by encryption but was instead protected by local access controls, the process is a little more complex. The edit task must be given access to the asset but that must be done at the point of storage. Changes can be made to access controls only when the location is known so it may[15] not be until the URL has been obtained from the resolver and the edit task has picked one if there are more than one, that any changes can be made.

CSAP Part 5: Implementation Considerations discusses key management in more depth.

5.6 Protecting The Integrity of Asset Management and Resolution

As we have stated, the goals in protecting the asset management and resolution are the same. In their simplest form, both are look-up tables with the asset management table being content addressable. With any table, there are four possible operations:

1. Add an entry
2. Remove an entry
3. Modify an entry
4. Look up an entry

We will return to look up in a moment and, for now, concentrate of the first three operations all of which can corrupt the integrity of the table. We refer to them collectively as “change the table.”

5.6.1 Change The Table Functions

Securing the change-the-table functions is a primary part of protecting the integrity of the workflow but it is important to understand what can be achieved.

Authorization has granularity in two dimensions. One dimension is the change-the-table functions which can be authorized individually or collectively. The other dimension is the part of the table that can be changed: the whole table, a subset (e.g., scope) of the table, sets of entries, or individual entries.

CSAP will limit access to authorized entities and if, and only if, it can discern the three change-the-table functions then it can manage them separately. The other dimension of authorization, what is being

[15] While the URL to access the asset isn’t known until returned by the resolver, it is likely that the location of the asset is known because, if for no other reason, the storage and access to it had to be provisioned.

changed, cannot practically be managed from outside asset management or the resolver without their participation.

From a risk perspective, the narrower the range of entries or the scope of the change-the-table functions, the smaller the attack surface. Therefore, more granularity is better.

The risks are:

Function	Example risks
Add	Entries referencing unsafe locations Entries for unsafe assets[16] Evil twin entries[17]
Remove	Loss of the location of an asset Disruption of an in-process workflow
Modify	Entries referencing unsafe locations Entries for unsafe assets Evil twin entries Disruption of an in-process workflow

5.6.2 Look Up

The security around look-up depends on the security requirements of the production. As mentioned earlier, it is the role of the policy enforcement point protecting the asset to prevent unauthorized access to the data. If the asset is secured appropriately, denying a look up request does not add to asset security since asset security should not rely on the attacker not being able to find an asset. However, if an attacker knows the location of an asset, they know where to attempt access to the asset. This is an important consideration with assets protected by access control lists since there are well known attack strategies that access data storage.

Knowing the location of an asset can also, for example, enable an attacker to launch a DDoS attack although resistance to DDoS cannot hinge on depriving the attacker of the location of an asset. The infrastructure should remediate that form of attack

Two underlying principles of CSAP, least privilege and zero-trust, tell us that only authorized look-up requests should be answered. Whether that is controlled within the asset management and the resolver, or in a CSAP Policy Enforcement Point associated with the resolver is a matter of implementation.

The granularity of the authorization at the resolver can be anything from authorization for all resolution requests to authorization for look up requests for specific table entries. The granularity is defined in

[16] An unsafe asset is, for example, an asset that can affect execution of an application with malicious intent. Obviously whether this is possible depends on what the assets are.

[17] An entry is introduced that has properties identical to a legitimate entry but is in some way malicious

authorization rules created by the authorization service, but it requires a security component within the asset management or resolver.