CSAP COMMON SECURITY ARCHITECTURE *for* PRODUCTION

SECURITY INTEROPERABILITY IN MEDIA CREATION

Contents

using independent business judgment. Each MovieLabs member company shall decide independently the extent to which it will utilize, or require adherence to, these recommendations. All questions on member company adoption or implementation must be directed independently to each member company.

1 Introduction

We are at an inflection point in the long history of media production.

As noted in the MovieLabs "Interoperability in Media Creation" white paper, creative teams are faced with historical levels of content production and more deliverables per title, along with increasingly complex workflows and rapidly changing technologies, and as always all under tight schedules with a scarcity of talent and resources. To help meet these challenges, we need to find ways to better support those teams in their creative workflows, while also giving them the flexibility to adopt new technologies with less friction and risk. Success will enable allow creatives more iterations to better realize their vision with the same resources and timelines.

Software is playing an increasing role in workflows, and as described in the MovieLabs 2030 Vision white papers,[1] media creation is increasingly produced on shared cloud infrastructure shared across the individuals, departments, and vendors working on a production.

While all this is happening, we are also faced with unprecedented security threats. Traditional perimeter-based security is no longer relevant as there are no boundaries in the cloud. As malicious actors launch highly sophisticated attacks, which increasingly leverage AI, we and other industries must evolve our view and practices around securing our content.

Cloud infrastructure security is complex.[2] While securing cloud infrastructure is difficult enough for enterprises where most users are employees and suppliers do not share the enterprise's cloud infrastructure, the problem is much more complex in media creation with emerging and future workflows using many external vendors and individual contributors.

> Traditional security is based around the idea that there is an 'inside' and an 'outside,' and that you can secure the inside with a secure perimeter. But the cloud doesn't have an inside and an outside, so what do you do?" *The Traditional Perimeter is Dead, Now What?* Oxford Computer Group, January 2017

A lack of security interoperability is a hidden threat. If the various security systems used in media creation workflows cannot act as one, exposed gaps will increase the likelihood of a breach and increase the extent of the breach, especially where it is unclear where security responsibility lies in complex interoperable, multi-vendor systems.

This MovieLabs interoperability paper details the case for interoperability in the software and systems that make up these workflows. The paper details the case for interoperability in the security of the software and systems.

---

[1] *The Evolution of Media Creation* (2019), *The Evolution of Production Security* (2020), *The Evolution of Production Workflows* (2020). https://www.movielabs.com/production-technology

[2] *Cloud Security Complexity*, Cloud Security Alliance, May 2019. https://cloudsecurityalliance.org/artifacts/cloud-security-complexity/

Sections 2-4 set out the milieu within which security components must interoperate. Specifically, this is the authorization policies, authorization and authentication in a zero trust network. We have chosen this order to describe them since authorization policies are the practical expression of authorization, and thus, it can be argued, the primary point of interoperability with authentication as a pre-requisite to authorization.

1.1 Common Security Architecture for Production

The 2030 Vision envisions an evolution of production into cloud workflows where the cloud infrastructure is shared by the production, the studio, their vendors, and users (both employees and individual contractors). This makes the security significantly more challenging than the enterprise infrastructure where most users are employees, and the infrastructure is not shared with other organizations.

The Common Security Architecture for Production (CSAP) is a collaboration-oriented Zero Trust Architecture (ZTA). It is concerned with securing and protecting the integrity of assets, processes, and workflows in the collaborative environment of media production. It is not concerned with protecting the underlying infrastructure, rather, it is designed to protect production on an infrastructure that is not trusted.

The fundamental rule of a zero trust architecture is that nothing is trusted until it has been authenticated. There is no concept of inside the security perimeter because there is no perimeter, nor is one needed. Anything can attach itself to the network, but that brings no assumption of trust. There are no trusted ports on the router: whatever is attached to the network cannot take part in any activity until it has been authenticated.

The CSAP Zero Trust Foundation is a zero trust architecture, as might be used in any enterprise, with certain characteristics that are part of the NIST Zero Trust Architecture – fundamentally that means security policies are used to authorize activity.

Two principles of the MovieLabs Common Security Architecture for Production are security by design and the requirement that security does not get in the way of the creative process. To achieve both, it is essential that the parts of the security system are interoperable.

1.2 Scope

The scope of this document is security control interoperability where the security architecture:

1. Is a zero trust system that aligns with the NIST Zero Trust Architecture
2. Has the features of the NIST Zero Trust Architecture that align with the CSAP zero trust foundation.

1.3 Reference and Informative Material

In this section, we present a set of reference and informational material from MovieLabs and government agencies in the US and UK that was used to develop this document and provides additional reading for those who want to delve deeper.

1.3.1 MovieLabs Publications

MovieLabs has produced a video explaining zero trust and why it is the security architecture of choice for protecting cloud production on both public and private cloud infrastructure:

> Zero Trust and Protecting Cloud Production (video), https://movielabs.com/zero-trust-and-protecting-cloud-production/

The MovieLabs Security Blog series is an introduction to the concepts behind zero trust and CSAP. At the time of publication, three posts have been published and there are more to come (https://movielabs.com/2030-vision-blog/). In order, the three posts are:

1. "Can I Trust You?" https://movielabs.com/can-i-trust-you/
2. "I Don't Trust You, You Don't Trust Me, Now What?" https://movielabs.com/i-dont-trust-you-you-dont-trust-me-now-what/
3. "Am I Authorized to do That?" https://movielabs.com/am-i-authorized-to-do-that/

The Common Security Architecture for Production (CSAP), a 5-part set of documents that describes the architecture and discusses implementation options. https://movielabs.com/production-technology/production-security/.

- Part 5A describes the CSAP Zero Trust Foundation. https://mc.movielabs.com/docs/security/part--implementation-considerations/part-a-starting-out/csap-recap/

Enhanced Content Protection for Production (ECPP), recommended practices for the transition from on premises to a hybrid or full cloud infrastructure, https://movielabs.com/prodtech/security/ML_ECPP_v1.0.zip

Ontology for Media Creation, https://movielabs.com/production-technology/ontology-for-media-creation/

1.3.2 Publications from US and UK Government Agencies

Zero Trust Architecture, US National Institute of Standards and Technology (NIST) Special Publication 800-207, https://doi.org/10.6028/NIST.SP.800-207

Zero Trust and Trusted Identity Management, US National Security Telecommunications Advisory Committee (NSTAC), https://www.cisa.gov/sites/default/files/publications/Final%20Draft%20NSTAC%20Report%20to%20the%20President%20on%20Zero%20Trust%20and%20Trusted%20Identity%20Management.pdf

Advancing Zero Trust Maturity Throughout the Network and Environment Pillar, US National Security Agency (NSA) Cybersecurity Information Sheet (CIS), https://media.defense.gov/2024/mar/05/2003405462/-1/-1/0/csi-zero-trust-network-environment-pillar.pdf

Zero Trust Architecture Design Principles, UK National Cyber Security Centre, https://www.ncsc.gov.uk/collection/zero-trust-architecture

Network Architectures, UK National Cyber Security Centre, (this article describes how zero trust improves security for remote access), https://www.ncsc.gov.uk/collection/device-security-guidance/infrastructure/network-architectures#section_2

1.4 Definition of Zero Trust Architecture

We define a zero trust architecture (ZTA) to be that described in National Institute of Standards and Technology (NIST) Special Publication 800-207 and that has the attributes necessary for it to be used for securing media production.

Zero trust (ZT) provides a collection of concepts and ideas designed to minimize uncertainty in enforcing accurate, least privilege per-request access decisions in information systems and services in the face of a network viewed as compromised. Zero trust architecture (ZTA) is an enterprise's cybersecurity plan that utilizes zero trust concepts and encompasses component relationships, workflow planning, and access policies. Therefore, a zero trust enterprise is the network infrastructure (physical and virtual) and operational policies that are in place for an enterprise as a product of a zero trust architecture plan. – NIST SP 800-207

The reader is reminded that neither SP 800-207 nor the MovieLabs Common Security Architecture for Production (CSAP) are deployment guides.

1.5 Notation

Since we are discussing security interoperability in the context of media production, we will use the term authorization policy (all lower case) to mean a security policy having the same attributes as a CSAP Authorization Policy. A CSAP Authorization Policy has the form of subject/verb/object sentence with some qualifiers:

Actor (Participant) can do something (Action) to an object (Asset) if some conditions (Application, Infrastructure, Timeframe) are met.

The Action may include other data or parameters.

In CSAP, Authorization Policies are an abstracted description of what is authorized and may be translated to Authorization Rules, the manifestations of Authorization Policies that are applicable to particular security infrastructures.[3] Authorization Rules are distributed to the appropriate Policy Enforcement Points.

In this document, our generic term "authorization policy" covers both CSAP Authorization Policies and Authorization Rules.

[3] To be pedantic: if the infrastructure understands an Authorization Policy in its native form, then the Authorization Rule is identical to the Authorization Policy.

2 Authorization Policy Flow

We start out our discussion on security interoperability with authorization policies. This core part of zero trust architecture is the element around which security interoperability can be constructed.

The NIST SP 800-207 describes an authorization policy (remember we are using the term authorization policies for NIST's security policies) as:

The set of access rules based on attributes that an organization assigns to a subject, data asset, or application. Environmental attributes may include such factors as requestor network location, time, reported active attacks, etc. These rules and attributes are based on the needs of the business process and acceptable level of risk. Resource access and action permission policies can vary based on the sensitivity of the resource/data.

Soon we will examine how control flows through a zero trust network from generation of authorization policies to the policy enforcement points, but first we need to describe a policy enforcement point (PEP) for those not familiar with the term.

2.1 Policy Enforcement Points

NIST SP 800-207 defines two architectural components that act on policies:

1. The Policy Decision Point determines whether the policy permits access.
2. The Policy Enforcement Point that sits between the untrusted and the resource.

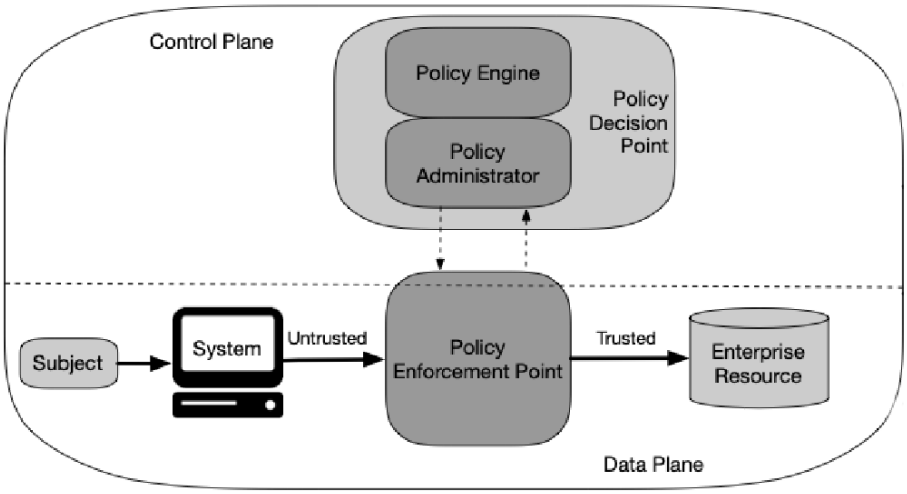

Figure 1 NP 800-207 Policy Enforcement [Source: SP 800-207]

While acknowledging this model, implementation of PEPs can take many forms as we will see shortly. The CSAP Authorization Rule, the infrastructure specific form of an abstract Authorization Policy, is designed to simplify evaluation of authorization policies. In CSAP, the functions of the PEP and policy decision point (PDP) are divided between the authorization service, which creates authorization policies, and what is referred to in this document as the PEP.

Ultimately, organization of the NIST model's PEP and PDP in a particular part of the infrastructure is an implementation choice strongly influenced by the infrastructure. This is because enforcement of an authorization policy, i.e., the role performed by the PEP, may be done by an infrastructure's native security controls and authorization policies are used to configure those security controls. Other influencing factors are latency and available compute resources for the PEP.

Regardless of how authorization policies are acted upon, these statements hold true:

1. Every resource must have a PEP controlling access to it.
2. A PEP is something an authorization policy is directly or indirectly pushed to.

Thus, there is always a layer where the PEP exists or its role is performed, and all access to a resource must pass through that layer. A group resources may use a single PEP.

2.2 Authorization Policies

We defined an authorization policy as being a machine readable expression of what should be authorized. The requirement for interoperability is that the PEP can read and interpret authorization policies. Requiring a user or a system administrator to configure a PEP or any access controls through a console or some other management tool does not support security interoperability.

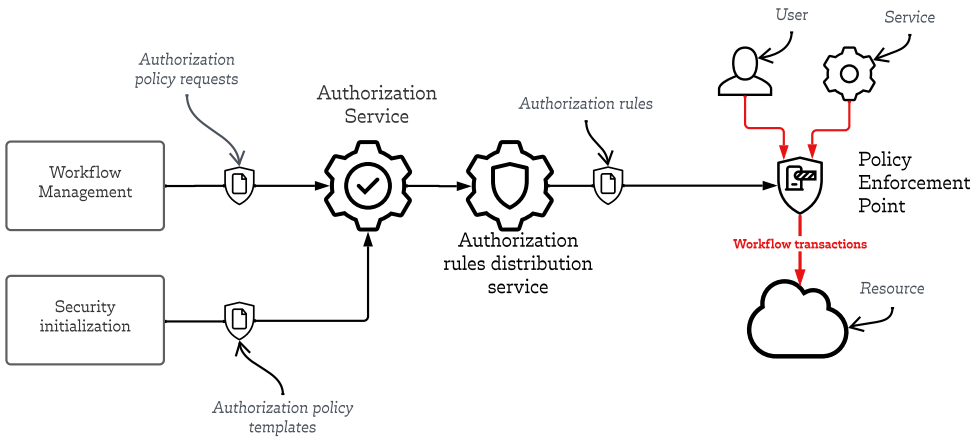

Figure 2 Flow of control in the CSAP ZTA

As ZTAs are controlled by security policies, it makes it easier to define the flow of control than in a security system without a defined architecture.

2.3 Interaction with Workflows

Workflow activities are the things that happen when a workflow is in operation. For example, a task uses data stored in cloud storage. It sends a read request to the storage to retrieve the data it needs. The policy enforcement point protecting the storage determines whether the read request is authorized.

The following diagram shows this example.

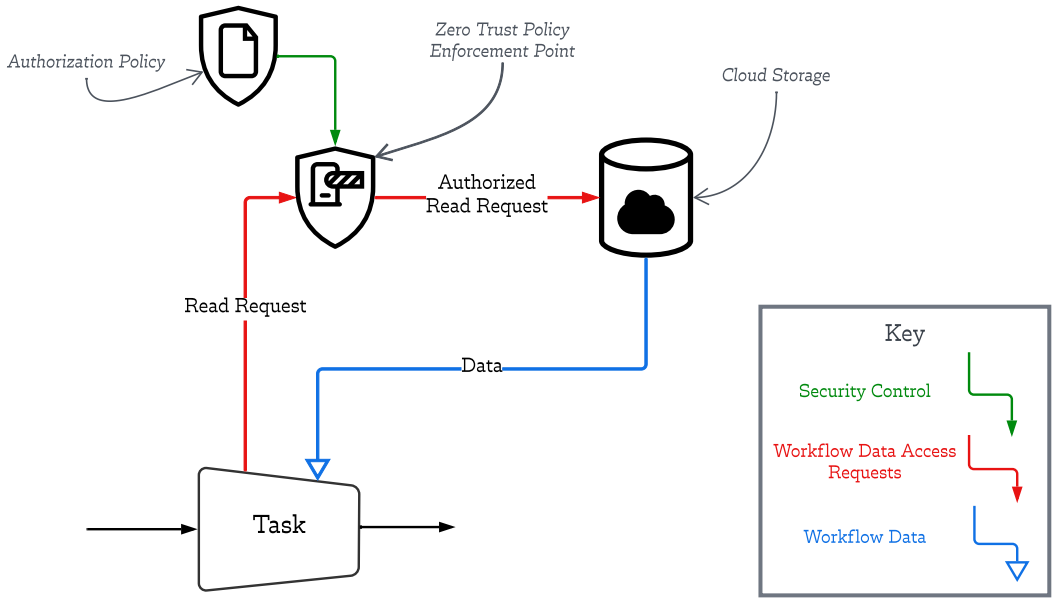

Figure 3 Example workflow

Not every part of the workflow activity directly interacts with the ZTA. For example, consider a messaging system where authorization policies control whether something can register to receive messages from a particular queue, but authorization policies are not used on a message by message basis because the messaging system is trusted to only deliver messages appropriately. The “register” function is an interaction with the ZTA, the routing of messages is not.

2.4 Authorization Policies and Security Interoperability

As we noted earlier, authorization policies are the lynch pin of security interoperability in a zero trust architecture. PEPs in the infrastructure can take on many forms, and each enforces the authorization policies directed to it. A part of the security infrastructure that cannot act on authorization policies directed to it can seriously inhibit security interoperability.

3 Authorization

As note, we are using the term authorization policy in this document to distinguish media production centric security policies from the more general meaning. An authorization policy is a machine readable expression of what should be authorized, and it can have a form like this:

Name	Definition	Defined by
Participant	Person, group, organization, or service.	Entity identifier
Action	What is to be done. Most commonly, a task.	
Application	Software with a user interface, software with API (e. g, a service)	Application name and version
Infrastructure	Piece of infrastructure, remote desktop on zero-client, server	Entity identifier, operating system, and platform
Timeframe	The period during which the authorization policy is in effect	Period delimited by time or events
Asset	File (asset, metadata), database entry, or group.	Asset identifier

How many of the parameters are used depends on the context. For example, participant should always be identified (although it could be “everyone” if security goals permitted that). For all of those parameters, granularity is determined by overall security goals and capabilities, all of which are part of risk management.

Security interoperability requires that an authorization policy in this form can be sent to a policy enforcement point and the authorized activity to take place. In zero trust, all activity not authorized by an authorization policy must be denied.

This document is about security interoperability, but it serves to remind the reader that a policy enforcement point must only accept authorization policies from a source that is both authenticated and authorized to be a source for authorization policies. The ZTA must itself be protected by authorization policies because the security of a ZTA breaks if there is an unauthorized source of authorization policies. Securing PEPs requires a "root of trust"[4] which can issue authorization policies to both PEPs and additional authorization policy issuers.

Obstacles to authorization policy interoperability are primarily associated with if, how, and where a policy enforcement point is deployed. A policy enforcement point does not necessarily need to be a new system component (see below).

[4] We use quotation marks around *root of trust* because we mean the concept of somewhere where security can be rooted rather than specific meanings such as a hardware root of trust as might be implemented in a Trusted Platform Module (TPM).

3.1 PEP Deployment Options

There are many ways that the Policy Enforcement Point (PEP) function can be deployed, and we present a non-exhaustive list of 6 ways.

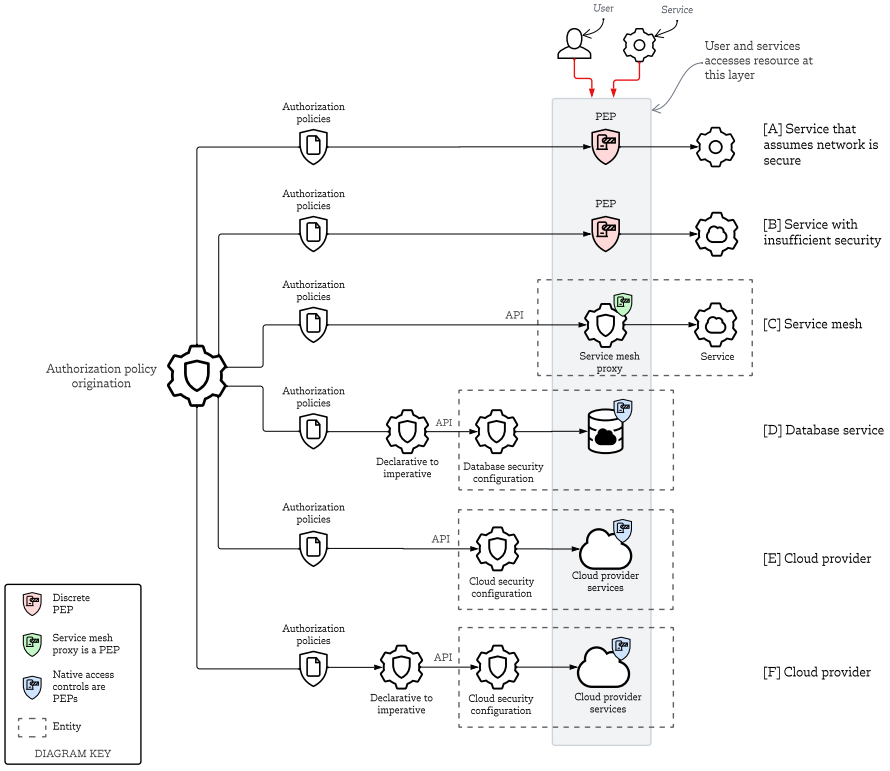

Figure 4 Different deployment options for PEPs

Figure 4 shows six ways that a PEP can be incorporated into a system.

[A] shows the deployment of a PEP in front of a service with no inherent security, for example one that was designed with the assumption that infrastructure it uses is secure, such as on-premises network protected by a security perimeter.

[B] shows the deployment of a PEP in front of a SaaS service that has insufficient security. “Insufficient” might mean, for example, that the SaaS service has security, but it does not have sufficient granularity or cannot be controlled by authorization policies because it uses a security administrator console.

[C] shows a service mesh. A service mesh is a secure way of deploying services that don’t inherently have their own security. Each service has a service mesh proxy and all data traffic to and from the service passes through the proxy. Since service meshes are controlled by policies, the proxy can be directly configured with authorization policies.

[D] shows the case of a database service, for example a MAM, where its native security serves as the PEP. The service has security configuration function that can be configured through an API. As with [F], the declarative form of the authorization policy must be translated to the imperative form of that API.

[E] shows a cloud provider where the cloud security configuration can be configured directly with authorization policies.

[F] shows a cloud provider where the authorization policies must be translated from a declarative form to an imperative form in order to configure the cloud's security.

In the first two, a policy enforcement point is required. In the last four, the policy enforcement point is within the security components of the infrastructure.

Interoperability depends on there being a place where the PEP exists, and that the PEP can receive and act on authorization policies that contain the appropriate parameters.

4 Authentication

Authentication is the cornerstone of zero trust.

A tenet of ZTA is that everything must be authenticated before it can be trusted. Everything includes users, devices and services and can be extended to other components such as software. Authentication means verifying that an entity is the trusted entity it is claiming to be.

When two things connect, the authentication must be mutual and happen at the same time as the connection is established and before any data is transferred.

The most common methods for authentication are an identity management system for users and a certificate authority for devices, services, and software.

For example, the encrypted TLS connection used for HTTPS access to a website such as a bank uses the website's certificate to authenticate that the website is indeed the website it claims to be. In that case, the website's certificate is issued by a public certificate authority and the device connecting to it should establish that the certificate is valid before initiating the encrypted connection.

Just as a user's authentication can be disabled at the identity management system, the certificate used to authenticate devices and services can be revoked by the certificate authority. That means that in both cases, proof should be sought that claims by a user, service, or device to be authenticated are valid at the time of use.

None of this is specific to media creation, nor are these the only methods available. To support interoperability in a zero trust environment, authentication must be supported in an interoperable way.

4.1 User Authentication

User authentication in zero trust is no different than elsewhere in IT security and much work has been done and is being done on authentication interoperability. Many standards and de facto standards exist.

For example, identity management is core to ZTA and there are many protocols that fulfil the role of the identity management. These include:

- Lightweight Directory Access Protocol (LDAP), which is an open-source protocol used in Microsoft's Active Directory,
- Security Assertion Markup Language (SAML), which is a protocol commonly used in Single Sign-On (SSO) applications,
- OpenID, which is less complicated than SAML and is used by, for example, web applications,
- OAuth, which is used by customer-facing platforms to connect third-party applications to a user's permissions to enable SSO,
- Kerberos, which is an open protocol using a system of tickets and authenticators to verify user identities,
- Remote Authentication Dial-In User Service (RADIUS), which is used for network connections such as VPNs,
- Diameter, which evolved from RADIUS and is now replacing it with a message-based authentication system,

- System for Cross-domain Identity Management (SCIM), which is an open standard capable of automating the exchange of identification data from one system to another, and
- Terminal Access Controller Access Control System (TACACS), which is a Cisco protocol developed for the US Department of Defense to simplify the process of authentication and authorization providing an SSO environment.

Zero trust does not require any specific choice of protocols, and the protocols used in a particular infrastructure are based on many factors outside of the scope of this document.

4.2 Identity and Access Management

Identity management (IDM), a core authentication component, can also have an access privileges or rights component. Identity management may include user attributes such as roles, group membership, and enumerated access privileges, and in such cases, it is called identity and access management (IAM).

An IAM system can provide both authentication and authorization. However, in ZTA, the two functions should be separable, at least conceptually. User identity in an identity management system is relatively static – typically the account is created when the user joins the organization and removed when they leave. However, authorization can be very short term. NIST SP 800-207 uses the term “dynamic security policy” for what we refer to as authorization policy, which emphasizes the ephemeral nature of authorization.

In both NIST SP 800-207 and CSAP, authentication and authorization are discrete functions. Zero trust means that just because something is authenticated does not mean it is authorized to conduct some activity or, indeed, any activity. Thus, an IAM system managing both authentication and managing access privileges is performing two discrete functions regardless of how the authentication and authorization data are delivered.

In the following diagram, authentication and authorization may be sent together but they are to be viewed separately.

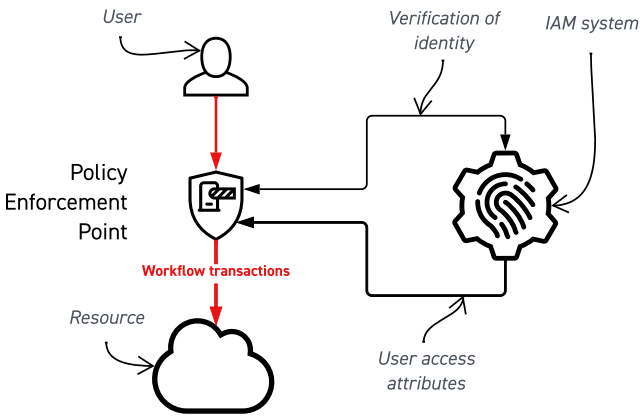

Figure 5 IAM transactions viewed logically

An IAM isn't the only way to implement authentication and authorization, and any solution that relies on there being a single source of authentication and authorization, such as an IAM, may require compensatory functionality to be fully interoperable. For example, using the System for Cross-domain Identity Management (SCIM) to set user attributes such as group membership. https://scim.cloud/#Overview.

5 Authorization Policy Handling & Interoperability

How authorization policies are handled is a core part of zero trust interoperability. It determines two things:

- If authorization can be set other than from an authorization policy, it is a potential barrier to successfully implementing zero trust.
- If authorization must be set other than from an authorization policy, it is a barrier to interoperability.

This means we can define a requirement for zero trust interoperability:

Authorization must be set by, and only by, an authorization policy.

Many systems are going to have security controls that can be used outside of a ZTA deployment. That is true of any cloud provider's infrastructure. That leads us to refine our definition of interoperability to be:

Interoperability is a property of a system as it is configured and deployed.

If a system is configured in a way that supports interoperability, then it is interoperable, however it isn't necessary for that to be the only way the system can be configured.

5.1 Discrete PEP

In Figure 4, [A] and [B] use a discrete PEP deployed in front of a resource.

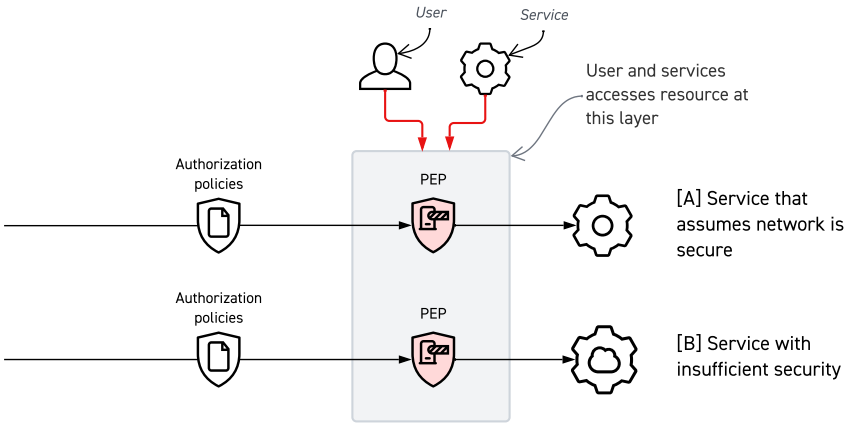

Figure 6 PEP Deployment options [A] and [B]

Interoperability is achieved if the PEP accepts and acts on authorization policies – if the PEP is created for this purpose, that is to be expected.

Discrete PEPs only work if:

1. All traffic to the service passes through the PEP.

2. The PEP has sufficient insight into the nature of the request and resources to enforce the required level of security granularity.

It is more obvious how to achieve the first of these. Solutions like the proxy in a service mesh or the network interface in a software-defined perimeter inevitably mean all traffic passes through the proxy in the first case, or onto the software-defined network in the second case.

The Cloud Security Alliance's Software-Defined Perimeter (SDP) is a cloud security architecture for protecting access to cloud resources. Membership of the SDP is managed by the SDP controller, and all communications within the SDP are carried over mutual TLS[5] connections managed by the SDP controller.

The SDP controller is a policy definition, verification, and decision mechanism – a Zero Trust Policy Decision Point – that maintains information about which identities (e.g., users and groups) from which devices should have access to an organization's services (on-premises or in the cloud). It determines which SDP Hosts can communicate with each other.

From the Software-Defined Perimeter Specification v2.0.

(Remember that our definition of a Policy Enforcement Point often encompasses the role of a Policy Decision Point.)

Implementing SDP on a resource means that all access other than an SDP connection is blocked.

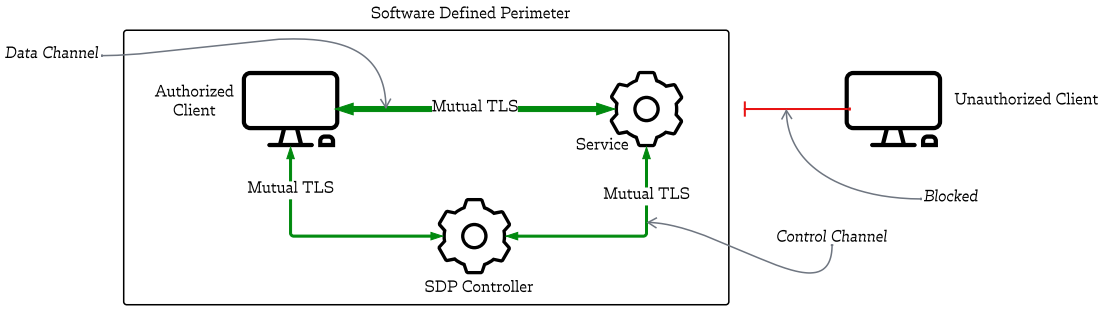

Figure 7 Software-Defined Perimeter

In the figure above, the unauthorized client cannot connect to the service even if it can be authenticated because the service only accepts connections that are authorized by the SDP controller, which means they come from authenticated nodes in the SDP.

However, that is not sufficient. The discrete PEP must have enough visibility into the nature of transactions to determine whether the activity is authorized.

[5] "The connections between all hosts must use TLS or Internet Key Exchange (IKE) with mutual authentication to validate the device as an authorized member of the SDP prior to further device validation and/or user authentication." Cloud Security Alliance SDP Specification 1.0

Consider [B] in Figure 6. The service holds assets in storage only it can access, and each asset has an identifier. A client of the service sends a read request for an asset, the read request carries an asset identifier which could be a file path or a URL as well as an abstract identifier.

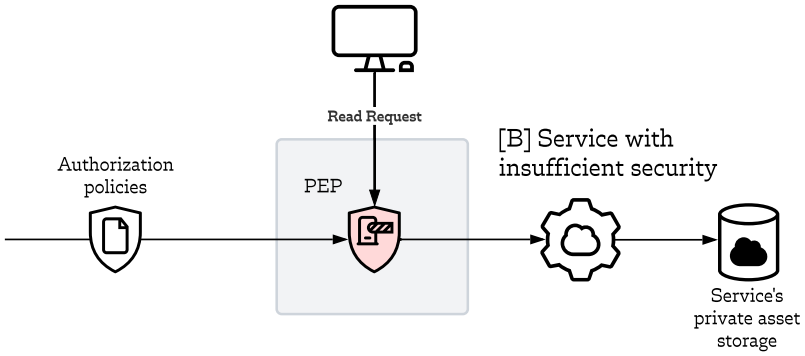

Figure 8 Accessing a service's assets

For the PEP to enforce authorization policies, it needs to know enough about the read request parameters to determine whether it is authorized.

The asset identifier in the read request must match the asset identifier in the authorization policy which ultimately means that whatever generated the authorization policy is using the same asset identifier as the service. The PEP must also understand the format of the transactions with the service to determine, in our example, that the message in question is a read request and what the parameters are, including the asset identifier.

In many ways, these are requirements general to workflow interoperability.

This means that a discrete PEP will not work for every service, and it is more than conceivable that services will exist that cannot be secured with an external PEP and also do not have a component that can be used as a PEP (as is the case in [D], [E], and [F]).

5.2 Service Mesh

Istio, a provider of service mesh technology, describes a service mesh in this way:

Modern applications are typically architected as distributed collections of microservices, with each collection of microservices performing some discrete business function. A service mesh is a dedicated infrastructure layer that you can add to your applications. It allows you to transparently add capabilities like observability, traffic management, and security, without adding them to your own code. The term

"service mesh" describes both the type of software you use to implement this pattern, and the security or network domain that is created when you use that software.[6]

As shown in Figure 4 [C], a service mesh proxy acts as a PEP and is controlled by authentication and authorization policies.

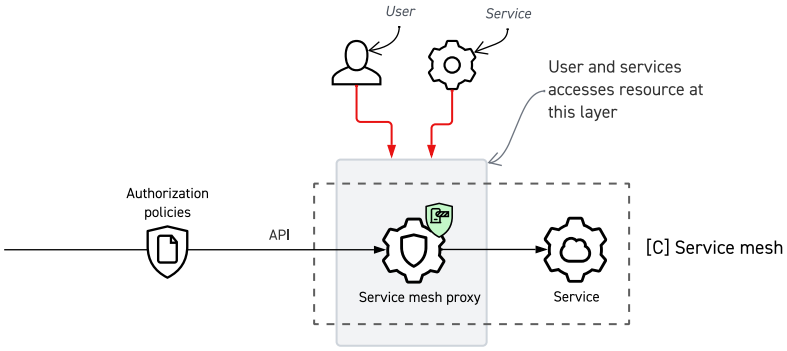

Figure 9 PEP Deployment option [C]

Provided that the ZTA can generate the appropriate authorization policies then the service mesh can be interoperable. This isn't a stretch goal since service mesh implementations use Open Policy Agent (OPA).[7] OPA is a policy engine that allows authorization policies to be applied to interactions within the mesh environment. OPA policies are described using Rego and authorization policies can be translated to Rego for use by the OPA.

5.3 Services with an Integrated PEP

In Figure 4, cases [D], [E] and [F] represent cases where the resource or service has an integrated security function that is managed via an API. The security function, for example Role Based Access Control (RBAC) in cloud storage, can act as the PEP if it can be programed using an authorization rule. For this to be the case, the security services must be capable of providing the required granularity of security in a way that can be controlled by authorization policies.

[6] https://istio.io/latest/about/service-mesh/
[7] https://www.openpolicyagent.org/docs/latest/

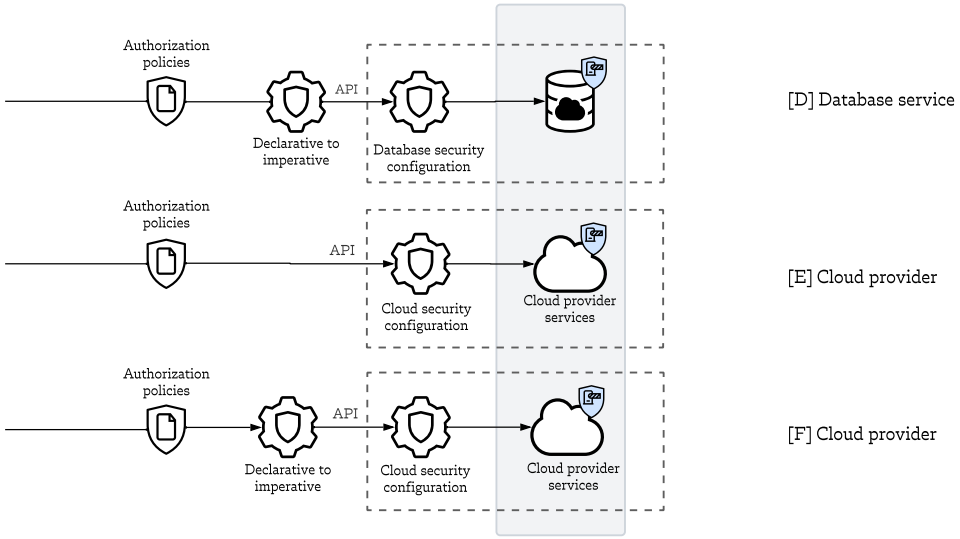

Figure 10 PEP Deployment options [D] and [E]

The difference between [D] and [F] on one hand and [E] on the other is in how the authorization policy is conveyed to the security service. In [E], an authorization policy is sent to the security service configuration which acts on it. [D] and [F] describe the cases where a component of the zero trust deployment receives the authorization policy and turns them into the imperative format used by the service.

[E] is likely to be more interoperable than [D] or [F] since an additional component is not required, however [E] is also less likely to be generally supported now but that is changing.

Although we describe these three cases terms of a database service and a cloud provider, they also apply to any infrastructure which has sufficient security services, for example, a SaaS provider with sufficient security or a storage system (cf. case [B], where that is not the case).

5.4 Interoperability Inhibitors

An inhibitor of security interoperability is the absence of any way to control integrated security programmatically[8] with the necessary granularity using authorization policies. (Programmatic control means that security management is fully achievable through machine-to-machine interfaces such as APIs.) There are two distinct components of that statement:

- A service has integrated security that can only be managed through an administrator console and cannot be managed by authorization policies.

[8] By saying "programmatically" we are saying that manual control of security does not count as interoperability.

- A service's integrated security can be managed through an API, but there is insufficient granularity in the integrated security services to support the granularity of the authorization policies.

We discussed the first of those points earlier, but it is worthwhile expanding on the second. Take for example a service that uses role-based access controls (RBAC) to control access. RBAC is a way to control access based on the roles assigned to users and the access permission assigned to role. It is possible to configure RBAC from an authorization policy but that only works if there is a sufficient level of granularity. If, in our example system, a user can only have one role and only one role can be set in each access permission, then it is very unlikely that any useful granularity is possible using roles. Now, if the access permissions could be set independently for each user, the necessary granularity can be set although possibly at the expense of complexity.

Practically speaking, anything that increases complexity above an acceptable threshold is an inhibitor of interoperability even if interoperability is still possible. The acceptable threshold would be defined by the implementor and, for example, might be defined by the effort needed vs. available resources, impact on ease of management, user experience, risk management, etc.

Another inhibitor of security interoperability is when the only option to use a service is with a discrete policy enforcement point as we have described earlier. Using the discrete policy enforcement to enable interoperability can be frustrated if:

- A service does not give enough visibility into the transactions between it and its clients to allow for authorization policies to be fully applied by a discrete policy enforcement point.

In Figure 4 [B], we described the case of a service that does not have sufficient security. If the insufficiency is in the granularity of the security controls and a discrete PEP cannot gain the necessary visibility into activity to make up for the insufficiency, the service will not be capable of interoperable working with the authorization system even with a discrete PEP.

Security interoperability means the ability to bring your own security to services such as SaaS offerings. Services that require the use of their own security mappings living in their applications are problematic since for interoperability we want the security to be controlled through programmatic interfaces or directly by authorization policies.

6 Workflow Management and Security Interoperability

Something must initiate the creation of authorization policies and in CSAP; workflow management is one of the initiators.

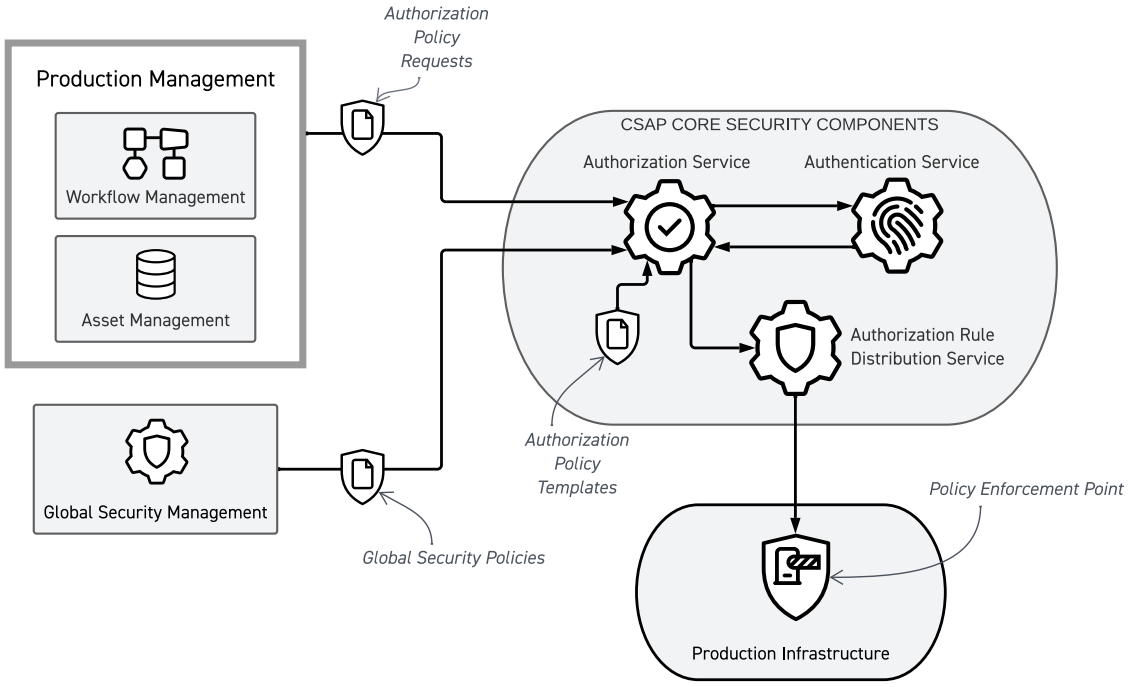

Figure 11 CSAP Core Components and Workflow Management

This is the start of the authorization policy life cycle.

The requirement for interoperability depends very much on how production is managed, especially how much automation takes place.

If workflow management is automated, security interoperability means that the automation system could create authorization policies. To accommodate this use case, products like SaaS services need to expose APIs to enable the external control of security policies.

Where the context requires it, the workflow management can create a dynamic authorization policy with a narrowly defined set of permissions and a minimum duration

If workflow management is not automated, then generation of authorization policy requests needs to be among the tasks that are carried out when a workflow, or part of a workflow, is provisioned or initiated.

Whether this is done when the workflow is initialized or as the workflow is executed depends, in part, on the granularity of the security controls.

7 Conclusion

In this document, we have discussed security interoperability in a zero trust security architecture. Core to zero trust is that everything must be authenticated before it is allowed to do anything, and all activity must be authorized by authorization policies.

There are two reasons to foster zero trust security interoperability. The first is that interoperability is a necessary part of enabling zero trust security which in turn is a necessary part of securing the new workflows of the 2030 Vision. CSAP is an architecture designed to support interoperable management of security but for that to be possible, it needs the security components to be interoperable.

The second is that in today's threat climate, security interoperability is a fundamental need for all cybersecurity systems and the absence of interoperability can be one of the root causes of a breach. In part, this is because a lack of interoperability means increased complexity. Complexity is the enemy of security.

This document has examined interoperability within a media creation infrastructure. We have described it in terms of interoperability for a zero trust architecture, of which CSAP is an example.

Some specific points have been made:

- Authentication interoperability is an issue that affects all security models, not just zero trust, and much work has been done and is being done on it. For that reason, it is not covered in depth in this document.
- Interoperability of zero trust authorization requires that authorization policies can control access to resources programmatically and do so with sufficient granularity.

Authorization policies are acted on by policy enforcement points deployed in the infrastructure. A PEP might be an existing security component of a system that can act as a policy enforcement point, or it might be a discrete policy enforcement point at, say, a service's interface. We have examined some of the ways that policy enforcement points can be deployed.

As zero trust architectures evolve, so do the policy enforcement points. Concepts like just-in-time access, dynamic authorization policies, and continual authorization are powerful tools for increasing system security and decreasing the scope of the damage[9] that could be caused by a security compromise in any single component. However, these only become useful once the foundation of a zero-trust architecture has been built. The focus of this paper has been on describing the tension between interoperability and security in a Zero Trust architecture; how that architecture may evolve and become more sophisticated is a topic for another time.

Much like interoperable software-defined workflows for media creation can't be considered without security, this security paper can't be considered without a broader understanding of the Common Security Architecture for Production (CSAP). Readers who are unfamiliar with CSAP should download all

[9] Often referred to as the "blast radius".

5 parts of the architecture documentation and consider it for their next system or workflow redesign by visiting www.movielabs.com/CSAP.