CSAP

COMMON SECURITY ARCHITECTURE *for* PRODUCTION

PART 1:

ARCHITECTURE DESCRIPTION

VERSION 1.1

Contents

1 Security Architecture Introduction

In September of 2019, MovieLabs published *The Evolution of Media Creation – A 10-Year Vision for the Future of Media Production, Post and Creative Technologies*[1] (the 2030 Vision), which foresees that within 10 years, and likely a lot sooner, all assets of a production will be stored in the cloud, and all processing of those assets will run in the cloud.

The follow-on work, *The Evolution of Production Security – Securing the 10-Year Vision for the Future of Media Production, Post and Creative Technologies,*[2] asserts that as production moves to the cloud and workflows transcend organizations, protecting cloud production requires a new approach to security.

This security architecture is a framework for implementing the security model described in *The Evolution of Production Security*. The architecture is presented in five parts:

Part 1: Architecture Description is this 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 Workflows discusses how the security architecture can be applied to software-defined workflows that are managed using a service bus.

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

Part 6: Policy Description Language

This document assumes that the reader has read *The Evolution of Production Security*.

Note that CSAP Part 1 version 1.1 removed the distinction between static and dynamic security policies since they behave the same and the difference is in implementation. Both are now simply referred to as "*authorization policies.*"

1.1 Goals of the Security Architecture

The goal of this document is to provide a reference architecture for those implementing (e.g., software tool vendors and service providers), constructing (e.g., studios and vendors), evaluating (e.g., auditors), managing (e.g., studios), and using systems for securing the assets and processes in cloud workflows for the production of movie and TV content.

This document describes the components of the security system and the interactions between those components. It strives to balance security, availability, usability, and cost-efficiency to deliver usable security.

CSAP is not a set of recommended practices for security. Every system component that is trusted must have a sufficient level of protection implemented to ensure that it is trustworthy. In some cases, CSAP

[1] https://movielabs.com/prodtech/ML-2030-Vision.pdf
[2] https://movielabs.com/prodtech/security/ML-Securing-the-Vision.pdf

and its Policy Enforcement Points may make that task easier, but how it is done and the level of robustness are matters for implementation based on, for example, organization requirements and risk analysis.

We are at the very beginning of production in the cloud and cybersecurity is a rapidly evolving field. Every effort has been made to describe an architecture that is flexible and durable; however, this is the first version. We will learn as implementation proceeds, and it is to be expected that the architecture will evolve, perhaps substantially.

1.2 Definition of Security

Our definition of security is:

- Protection from malicious and unauthorized activity, such as the exfiltration of assets.
- Protection of the integrity of data, workflows, applications, and processes.

Unauthorized activity includes, for example, activity by an unauthorized user and unauthorized activity by an authorized user. What appears to be an authorized user may be an intruder who has acquired the credentials of an authorized user.

The objects that need to be protected fall into three categories:

1. Assets: Data and metadata that are created, processed, and output.
2. Processes: Software services and user-interacting applications that process assets (including automated tasks).
3. Workflows: Orchestrated sets of processes acting on a set of assets.

The security threats to production are not solely the theft of assets. Simply protecting assets is not enough. Applications must be protected to ensure that their function is not subverted, or their output redirected. Only participants, whether that be a user or a vendor, authorized to work on a scene can be allowed to do so. If it is important to a production that a particular version of an application is used, that must be what happens.

The security architecture integrates with software-defined workflows, which use a highly configurable set of tools and processes to support creative tasks by connecting them through software-mediated collaboration and automation. More Information on software-defined workflows, and associated concepts such as participant and task, can be found In the MovieLabs white paper *The Evolution of Production Workflows – Empowering Creative Processes with Software-Defined Workflows.*[3]

1.3 The Role of Security in Production

The security architecture described in this document is designed for the paradigms of production in the cloud described in the 2030 Vision, which means any internet-accessible compute and storage infrastructure that uses scalable cloud technologies, regardless as to whether it is operated by a

[3] https://movielabs.com/production-technology/sdw/

hyperscale cloud provider, in a private data center, on premises, or some combination thereof. This security architecture can be applied to any production using cloud technology.

Today, the role of production security is to address risks that are common to many businesses. The goals of security are to maintain:

- Confidentiality by preventing unauthorized disclosure, typically through the egress of data,
- Information integrity by preventing modification or destruction of data,
- Availability by preventing disruption of information or system access or use.

These risks are mitigated through security resiliency. For example:

Threat	**Preparedness Strategy**
Vandalism, misconduct	Basic security hygiene
Incursion/abuse	Critical information protection
Presence, breach	Responsive awareness
Disruptions, espionage	Cyber resilience
Cyber conflict, warfare	Pervasive agility

(Credit: Roin Nance, University of California, Berkeley)

However, *The Evolution of Production Security* paper adds another role for production security: protecting the integrity of workflows by ensuring a workflow is conducted as intended, using:

- Approved participants whether human or machine,
- Approved/designated applications,
- Approved systems.

2 Document Conventions and Concepts

2.1 Definitions

Within this document, we use certain words to represent a broader category of similar items.

Application	Software that performs an operation that is part of a production workflow. The term includes processes that operate semi-autonomously or autonomously.
Application Image	An *application image* running on a *device* where the software and the device are indivisible or running in a serverless environment, that provides a set of functions to clients. The application image provides one or more services or microservices. We regard this as an atomic unit from the security point of view.
Artifact	Anything on the network used by the production that is not a human. That includes resources, and assets, and network components that are not part of the workflow.
Attack Surface	This is the sum of the different points, attack vectors, where an attacker can try to breach the security of a system.
Device	A hardware system or a virtualized hardware system with similar characteristics to a hardware system. Unless otherwise stated, no distinction is made between a physical server running Linux in a data center or a virtual machine running Linux on a cloud service.
Entity	Anything that can be authenticated.
Resource	A broad term used for anything that does work as part of a production workflow that is not a human. For clarity, *device*, *application,* and *application image* are resources.
User	A participant (see below) that is an individual person.

Other terms are defined in the broader MovieLabs Cloud Production work. These are:

Participant	The people, organizations, or services that do work related to a production. Participants have roles associated with them that define the precise nature of their work and can be related to entities such as Tasks and Assets.
Asset[4]	The data and metadata that are created, processed, and output by Tasks during a production. The architecture does not draw any distinction in terms of format or purpose. An asset is typically a file.

The acronyms used in this document include:

[4] This document references NIST published documents, in particular Special Publication 800-207, Zero Trust Architecture. To avoid confusion, when consulting the NIST documents be aware that the word “asset” in NIST documents is the same as the word “resource” in this document.

AES	Advanced Encryption Standard
DRM	Digital Rights Management
ID	Identifier/Identity
ISO	International Organization for Standardization
NIST	National Institute of Science and Technology
SaaS	Software as a Service
VFX	Visual effects
VPN	Virtual Private Network

2.2 Scalability

Risk assessment is a combination of the likelihood of an event happening and the consequences of it doing so. When combined with the cost of mitigation, we get an expression of risk tolerance. Understanding risk tolerance allows decisions to be made about which risks will be mitigated and to what extent. However well the security is designed and implemented, greater security is typically more expensive. Whether formal or informal, the outcome of a risk management[5] process is a guide as to how robust the security needs to be – the scaling of security.

This security architecture enables the security to be scaled to accommodate a production's risk tolerance and security budget.

Part 3 of the security architecture uses the construct of security levels to illustrate how security can be scaled in an ordered manner.

[5] There are many accepted methods of assessing risk, such as ISO 31000:2018,[5] Risk Management Guidelines, https://www.iso.org/obp/ui#iso:std:iso:31000:ed-2:v1:en.

3 Foundational Concepts

The MovieLabs 2030 Vision postulates that cloud services are common resources shared across everyone engaged in a production, be that the production company, the studio, VFX houses, finishing houses, small specialist providers and individual contributors. That presents a very different security challenge from today's use of cloud technology where the infrastructure is a hybrid cloud or data center within the control of a single entity and where few of those authorized to access the infrastructure are not employees.

In this section, we look at some important concepts that form the foundation of the security architecture.

> The enterprise perimeter is no longer a location; it is a set of dynamic edge capabilities delivered when needed as a service from the cloud.
>
> *The Future of Network Security Is in the Cloud, Neil MacDonald, Lawrence Orans, Joe Skorupa, Gartner, August 30, 2019*

3.1 Zero-Trust Architecture

This security architecture is a Zero-Trust Architecture[6] (ZTA), which starts with the belief that nothing should be automatically trusted either inside or outside of any security perimeter. Instead, the rule is to verify anything and everything trying to connect before granting access.

In August 2020, NIST published SP 800-207, Zero Trust Architecture.[7] That document presents these tenets of the zero-trust architecture:

1. All data sources and computing services are considered resources.
2. All communication is secured regardless of network location. Network location alone does not imply trust.
3. Access to individual enterprise resources is granted on a per-session basis. Trust in the requester is evaluated before the access is granted, and it is granted with the least privileges needed to complete the task.
4. Access to resources is determined by authorization policy. This includes the observable state of client identity, the application or service, and the requesting resource. It may include other behavioral and environmental attributes.
5. The enterprise monitors and measures the integrity and security posture of all owned and associated resources. No resource is inherently trusted. The enterprise evaluates the security posture of the resource when evaluating a request.
6. All resource authentication and authorization are dynamic and strictly enforced before access is allowed. This is a constant cycle of obtaining access, scanning, and assessing threats, adapting, and continually reevaluating trust in ongoing communication.

[6] Zero Trust Networks: Building Secure Systems in Untrusted Networks, O'Reilly, Evan Gilmar, Doug Barth ISBN-13: 978-1491962190

[7] National Institute of Standards and Technology, Zero Trust Architecture, NIST Special Publication 800-207, Abstract: https://csrc.nist.gov/publications/detail/sp/800-207/final, PDF: https://doi.org/10.6028/NIST.SP.800-207.

7. The enterprise collects as much information as possible about the current state of assets, network infrastructure, and communications and uses it to improve its security posture. This data can also be used to provide context for access requests from subjects.

Our architecture is built on these tenets.

3.2 Extrinsic vs. Intrinsic Security

The figure below shows the elements of a simple task. Today it would be carried out on a trusted infrastructure within the confines of a facility's security perimeter.

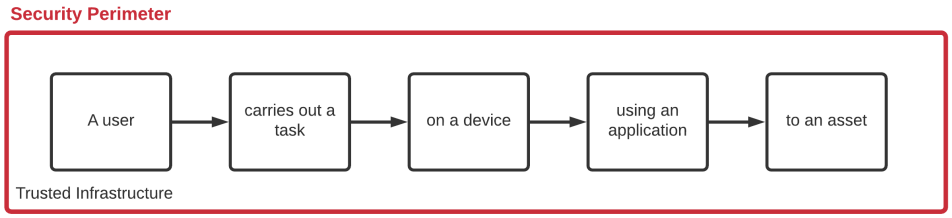

Figure 3-1 Extrinsic security

The security is extrinsic, meaning security is not designed into the elements, and to be secure, the work must be carried out within the confines of security imposed from outside of the workflow.

With production in the cloud, the cloud is a common set of resources across a production. It is used by the production itself, and by vendors, and by individual contributors. An all-encompassing security perimeter would be extremely complex, difficult (if not impossible) to manage, and would impinge on the creative process, making the security perimeter an undesirable option.

The solution is to create an intrinsically secure workflow like this:

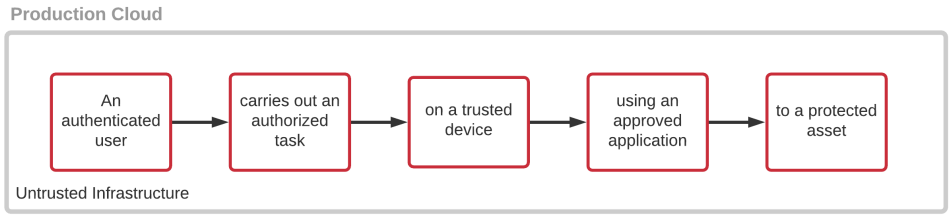

Figure 3-2 Intrinsic security

Here, security is built into everything that makes up the workflow. It is designed to be secure on an untrusted infrastructure. To accomplish that, we use a zero-trust architecture.

3.3 Trust

To be authenticated, something must be trusted so we must first have a common understanding of what trust means.

Mayer, Davis, and Schoorman (1995)[8] define trust as the willingness of a party to be vulnerable to the actions of another party based on the expectation that the other party will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party. This is an excellent definition for our purposes because it hints at the consequences of trusting something that is not trustworthy.

Anything that is to be trusted must prove it is the trusted entity it claims to be.

3.4 Authentication vs. Authorization

Authentication and authorization are the foundational concepts of the security architecture. They provide the answers to the questions:

- Who are you?
- Are you allowed to do this?

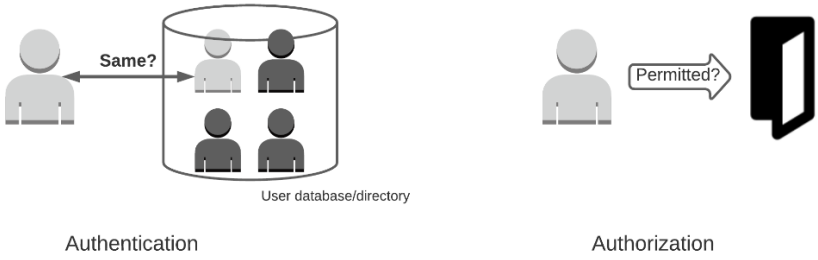

Figure 3-3 Authentication and Authorization

In this architecture, authentication and authorization are treated as separate functions. The reason for this will become clear when we address how the authorization service works.

Authentication is the process of determining that someone or something is what it purports to be. For example:

- Is this user ID being used by the person it belongs to?
- Is this system the one it appears to be to the user?
- Is this application unmodified and the correct version?

Authorization is the process of determining whether something, an action, is permitted. Once authentication has been completed, authorization addresses the questions:

- Is this user authorized to log into this system?
- Is this system authorized to run this service?
- Is this combination of user, device, and application authorized to access this asset?

[8] Mayer, R.C., Davis, J.H., Schoorman, F.D., 1995. An integrative model of organizational trust. Academy of Management Review 20 (3), 709–734.

Access controls are only part of the function of the authorization service. Access controls determine whether a user is allowed to open a file, make use of a SaaS application, etc. As the name says, they are used to control access to something.

Conventional access controls are only a part of authorization: authorization asks whether the activity being undertaken is to be permitted given the full context of the activity. For example, can this user run that application on that device at this time? In that question, there are three specific elements to the context.

Authorization is a property of the activity, whereas access control is a property of an artifact such as a system on a network or a file on a file system.

Finally, the distinction between authentication and authorization should not be taken to mean that the architecture cannot be built using an identity and access management (IAM) system. Implementation is outside the scope of this document.

4 High-Level View of the Architecture

This is a collaboration-oriented zero-trust security architecture. 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, and as discussed in Section 3.2, it is designed to protect production on an infrastructure that is not trusted.[9]

This architecture creates workflows with intrinsic security.

The security system authorizes activity at the direction of production management, a term used here to describe the point(s) of control of production activity.

The security architecture is designed around a control plane and a data plane. The control plane is divided into three sets of components: core security components, supporting security components, and production management. It provides security functions such as security decision making, messages that enable or deny activity, and the services required to make that operate.

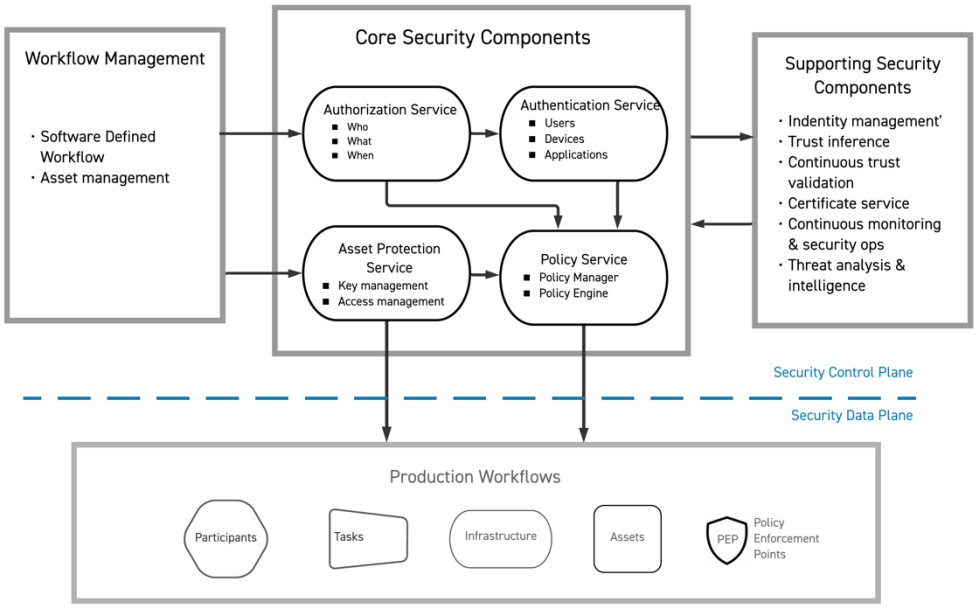

Figure 4-1 The high-level architecture

The data plane is where the security policies created by the core components are acted upon. It is embedded in the operation of the production cloud. The security data plane includes the workflow management discussed in detail in "Part 4: Security and the Software-Defined Workflow."

The distinction between core components and supporting components is based on the need for domain knowledge of the application.

[9] This does not mean that perimeter security cannot be used. An implementation could make use of a security perimeter, a software-defined perimeter (SDP), or microsegmentation.

The core components need domain knowledge of the application; they act in a way that is specific to securing production.

- *Authentication Service*. Identity is at the core of the architecture; the authentication service authenticates users, devices, and applications.[10]
- *Authorization Service*. Driven by workflow management, this service enables access to resources and assets according to the production's chosen security profile.
- *Asset Protection Service*. This service enforces authorized access to protected assets.
- *Policy Service*. Combining authentication, authorization, and production security policies, this service programs the policy engine described in Section 5.4.2.

The supporting components do not require application domain knowledge to provide the services used by the core components. It is expected that the secondary components can be implemented using off-the-shelf security components, including those provided by the hyperscale cloud providers and by vendors offering security-as-a-service.

- *Identity Management*
- *Trust Inference*
- *Continuous Trust Validation*
- *Certificate Service*
- *Continuous Monitoring and Security Operations*
- *Threat Analysis and Intelligence*

The two primary components of production management that are integrated with the security architecture are:

- *Workflow Management*. This initiates tasks, specifying what is to be done, where it is to be done, and who or what is to do it. In its simplest form, it is a work scheduling system generating work orders. It could be an orchestration system.
- *Asset Management*. This identifies the assets required for a task and the means to locate the asset.

The architecture can be implemented as a distributed system with multiple instantiations of the core security component services. There is no inherent design decision that prevents such an implementation, although additional design work may be needed to coordinate a distributed system. It is expected that there will be multiple workflow management and asset management systems.

The next sections look at these components in more detail, starting with the core security components and then looking at the supporting security components.

[10] The Authentication Service is not necessarily a server engaged in every authentication verification. For example, authentication can be carried out in the identity management in the supporting security components.

5 Core Security Components

Security is controlled by the four core security components.

These are:

- The authentication service,
- The authorization service,
- The policy service, and
- The asset protection service.

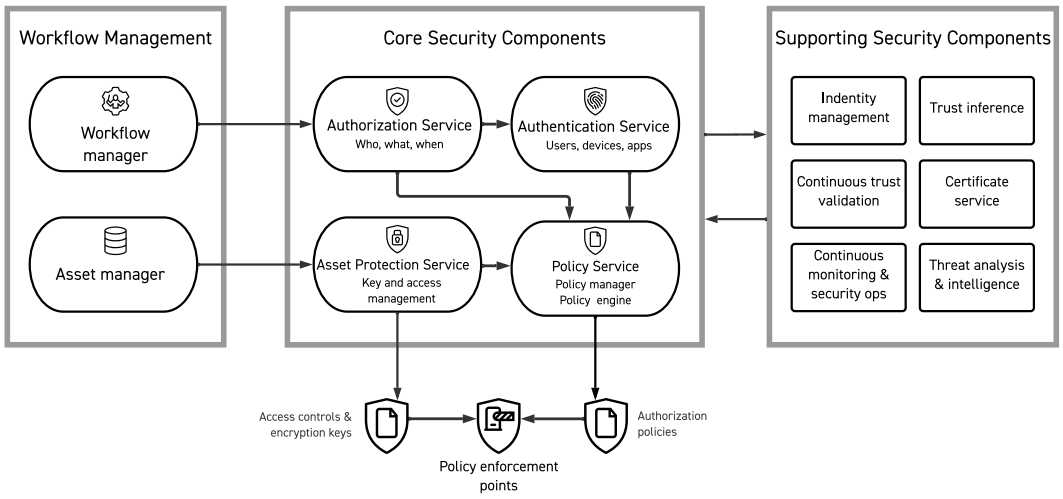

Figure 5-1 Detail of Core Components

Production activities are secured by the authentication and authorization services acting in concert to enable approved activity. If we go back to our simplistic view of a secure task, and remembering that this is a conceptual example, the authentication and authorization services act as follows:

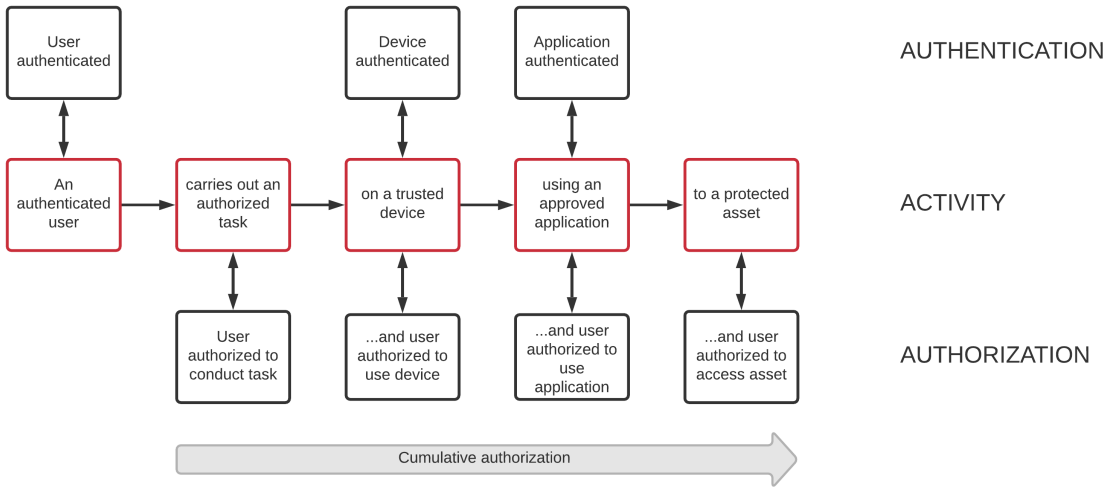

Figure 5-2 Authentication and authorization enable secure activity

When generalized, the "user" in Figure 5-2 would be a "participant" as defined in the MovieLabs publication *The Evolution of Production Workflows – Empowering Creative Processes with Software-Defined Workflows.*[11]

5.1 Authentication Service

At the very heart of the architecture, and any ZTA, is identity. Nothing is permitted to happen without every involved entity being authenticated.

Every participant (e.g., users, application services, etc.), application, and device has an identity, and the identity has attributes assigned to that account or resource. Attributes can include device characteristics such as software versions installed, network location, time/date of request, previously observed behavior, and installed credentials; it can include application characteristics such as version and signature. Once authenticated, these characteristics are trusted.

The authentication service is a policy-driven service responsible for authenticating and maintaining the authentication of the identities of participants, devices, and applications. The authentication service policies determine how it authenticates entities and how that authentication is managed over time.

It uses four of the supporting security components.

- Identity management and trust inference are used when first establishing or re-establishing trust. The balance between using only identity management, which implies a full credential check, and using the trust score from the trust inference service is a policy set by security administrators. The goal is to reduce the overhead of authenticating entities, particularly users.
- Certificate management provides the certificates and other cryptographic elements used for the creation of tokens and identification of entities, particularly ones that are not users.
- Continuous trust validation supplies the authentication service with alerts when the context of the original authentication changes and lowers the trust score.

5.2 Authorization Service

The authorization service is a policy-based service that authorizes an activity to take place. Specifically, it authorizes the elements of the workflow to participate and access the required assets. The authorization decision may occur at the time that the activity happens, or it may occur before. Preferably, the workflow management system notifies the authorization service of the activity to be carried out ahead of time, and the process of authorization does not introduce any latency to the workflow.

An authorization decision creates a n authorization policy that is passed on to the policy service.

All authorization policies start with a policy template. The template reflects the security requirements of the production derived from risk management, including risk tolerance and cost along with other parameters specific to the workflow and stage of production.

Authorization policies are:

[11] https://movielabs.com/production-technology/sdw/

- Constructed from access permissions based on attributes assigned to a user or device, to the asset, etc., and from environmental rules based on attributes such as network location, time, etc. They can be applied to resources and assets using common access control methods such as ACLs. These policies implement the principle of least privilege spatially.

  Policies constructed in this way are often preconfigured, although that does not mean they are locked. They are updated when events occur, such as the addition of a new user or the onboarding of a VFX vendor. Role-based permissions would normally be used to create this type of authorization policy.

- Constructed in response to instantaneous operational requirements, e.g., work being scheduled in a workflow. Policies constructed in this way may last no longer than the duration of an activity and cover no more than the resources involved in a workflow. These policies implement the principle of least privilege spatially and temporally.

There is a range of granularities possible for authorization policies. For example, authorization to provide access to assets might range from authorization to access all of a production's assets to authorization to access just the assets required for the immediate task for the duration of that task.

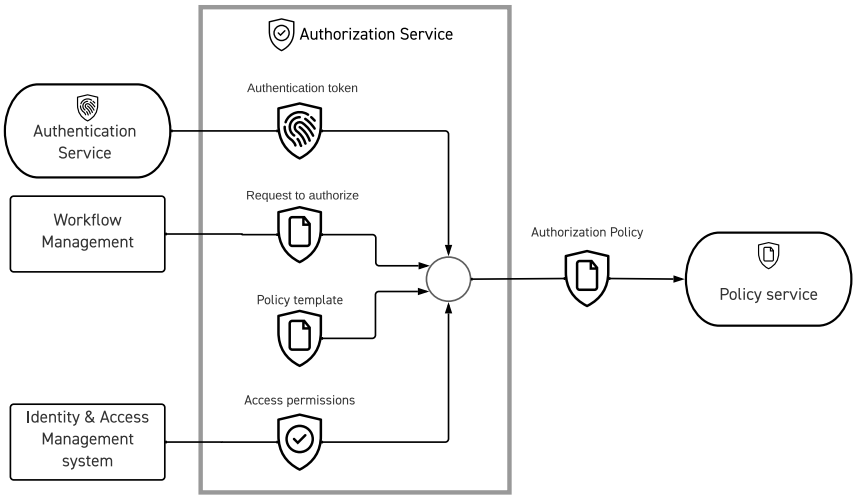

Figure 5-3 Authorization service operation

For example, the production management system at a VFX company assigns a VFX artist to work on a particular shot to be carried out over the next four days, and the authorization service is notified of the assignment. Depending on the need, policies can be constructed that:

- Permit the user to conduct the activity because the user is a member of a group permitted to conduct the activity.
- Permit the user to conduct the activity only during the time that the user is scheduled to do it.

Since a ZTA is built on the principle that only authorized activity is permitted, there is no need for the notion of "forbid" in policies since CSAP is deny by default.

Authorization for the participant to conduct the activity could be a result of the application of both of those policies.

- The participant is a member of a group permitted to conduct the activity AND the user has been assigned the activity.

In this case, authorization will be granted to the user who has been scheduled by the production management system (expressed in the authorization policy), provided that the user is permitted to conduct that activity in general.

Constructing authorization policies in the ways described here is part of the way that the security can be scaled so that the cost of the security is appropriate.

The use of policies in the security architecture will can be defined using the Policy Description Language.

5.3 Asset Protection Service

Traditionally the way assets are protected is by controlling access to the files and storing them on encrypted media. *The Evolution of Production Security* paper points out that the security goal is to protect the asset and not the storage container in which the asset is stored.

This architecture can protect assets individually. Controlling who can access an asset is a better security proposition than controlling who can access the storage. Controlling access at the asset level means least privilege – users are only granted access to the assets they need. Controlling access at the storage level means that some users are granted access to the assets they need and other assets in the storage container which they do not need. This is true regardless of the type of storage and network, although the implementation may be dependent on the type.

Two primary methods of protecting assets are:

- The use of the access permissions supported by the infrastructure (file system, cloud provider, etc.),
- Encrypting the assets with individual keys.

These are not mutually exclusive, and assets can be protected in different ways along their journey.

5.3.1 Asset Security using Access Permissions

Controlling access to resources using access permissions is a well understood discipline, and there are many ways to implement them.

One approach is the use of access-control lists (ACL). An ACL is a data structure containing entries that specify individual and groups rights to specific artifacts. It is an artifact-based access control.

The simplest example is the access control attributes of a Unix file where rights to read, write and execute can be assigned to a single user, a single group of users, or everyone. More generally, this is a

mechanism that implements access control for a system resource by enumerating the identities of the system entities that are permitted to access the resources.[12]

Another approach is using role-based access control (RBAC). Role permissions may be inherited through a role hierarchy and typically reflect the permissions needed to perform defined functions within an organization. A given role may apply to a single individual or to several individuals.[13] RBAC is regarded as "higher" level method since it ties permissions to specific operations with significance to the organization.

A third approach is attribute-based access control (ABAC), an access control paradigm whereby access rights are granted to users through the use of policies that combine attributes together. The policies can use any type of attributes (user attributes, resource attributes, environment attribute etc.).[14]

If access controls are used to protect individual assets, they will be implemented in the as part of an access control system that is part of the infrastructure, whether that be the file system access controls or full-functioned access management from a cloud provider.

The requirement is that only access authorized by the authorization service is possible. Conversely, there should not be any roles such as administrators or superusers that can circumvent the access controls or change permissions. That requires informed management of administrator accounts.

5.3.2 Asset Security using Asset Encryption

When asset encryption is used, granularity can be adjusted using strategies such as an individual encryption key for each asset or an encryption key for a group of assets. (In this latter case, whether each asset is encrypted separately using the same key, or the assets are encrypted as one group, is an implementation choice that may not materially affect security.)

The process of granting access starts with the identification of an asset that is needed. That happens when a user or process attempts to access the asset (on demand) or when an activity is scheduled that uses that asset (pre-loading).

The term "encryption at rest" is often used to refer to storage encryption as distinct from encryption during transfers, often referred to as "encryption in transit." Encryption at rest is imprecise. For the purposes of CSAP, we categorize encryption of stored assets we define two classes, which are not mutually exclusive:

- **Implicit encryption**. We define implicit encryption to mean that whatever is holding the asset (a storage "container,"[15] such as a disk, or a filesystem volume) is encrypted. Typically, the encryption is a property of the infrastructure; the container is encrypted as a property of the storage mechanism.

[12] NIST SP 800-82 Rev. 2
[13] NIST SP 800-53 Rev. 4
[14] NIST SP 800-192
[15] Including object storage, file storage, block storage, volume storage, hard drives…

- **Explicit encryption.** We define explicit encryption to mean assets are encrypted individually or as a group such that the encryption is independent of how the assets are held. We refer to this as "asset encryption." It is also referred to as "file encryption."

Access to assets protected by implicit encryption is controlled primarily by access controls. It protects only from an attacker who does not have access privileges (or cannot circumvent them) but has direct access the storage medium (e.g., an attacker who steals a laptop and removes the hard drive). However, access to assets protected by explicit encryption is controlled by selective distribution of encryption keys. Any form of explicit asset encryption requires a key distribution service. Where implicit encryption is used in a workflow today, assets are encrypted at certain points in their journey: they are stored in encrypted containers (e.g., encryption at rest or hard drive encryption) and transferred by encrypted file transfer protocols. As the asset moves around, it goes through a series of encryption/decryption cycles as shown in the following diagram.

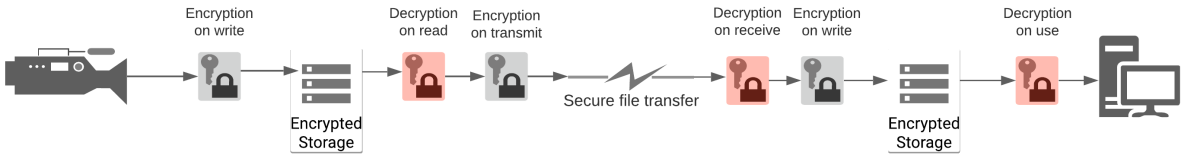

Figure 5-4 Encryption/decryption cycles of point encryption

However, explicit asset encryption supports end-to-end encryption. An asset is encrypted when it is created and is only decrypted on use. If we apply asset encryption in the diagram above, it looks like this:

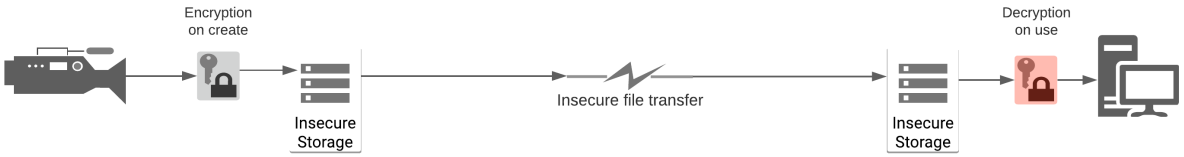

Figure 5-5 End-to-end asset encryption

Two things are obvious. The latter process is simpler, there are no special requirements on the storage or file transfer and there are fewer attack surfaces – each point where the data is unencrypted has an attack surface.

If access to assets is controlled by encrypting the assets, there is there is little reason to protect them further unless you believe the encryption system will be broken during the value lifespan of the assets.[16] Under the assumption that it will not, what needs to be protected is the encryption key, a small amount of data that is easy to move around and easier to secure than any asset file. The encrypted files can be

[16] Scott Fluhrer, "Reassessing Grover's Algorithm," IACR Cryptology ePrint, August 27, 2017, https://eprint.iacr.org/2017/811, posits that quantum computing will not break AES.

visible to all and can be managed (for example, copied) by anyone, but only those authorized can open the encrypted file to get access to the clear contents.[17]

The key protecting the content should be accessible only to dedicated software-based or hardware-based trusted components.[18] For example:

- The trusted component could be in the application. If so, it might be a small piece of secure code used at the point that assets are read and written.
- The trusted component could sit between the application and its storage. If so, it might be a wrapper through which assets are read and written, or it might be a hardware component in the file access used by the virtual machine on which the application is running.

This form of asset encryption is routinely used in the distribution of content to consumers, and at the content encryption level there is a similarity between how DRM content encryption works and what is needed for asset encryption. (DRMs also convey rights information, but that function is not relevant to this discussion.)

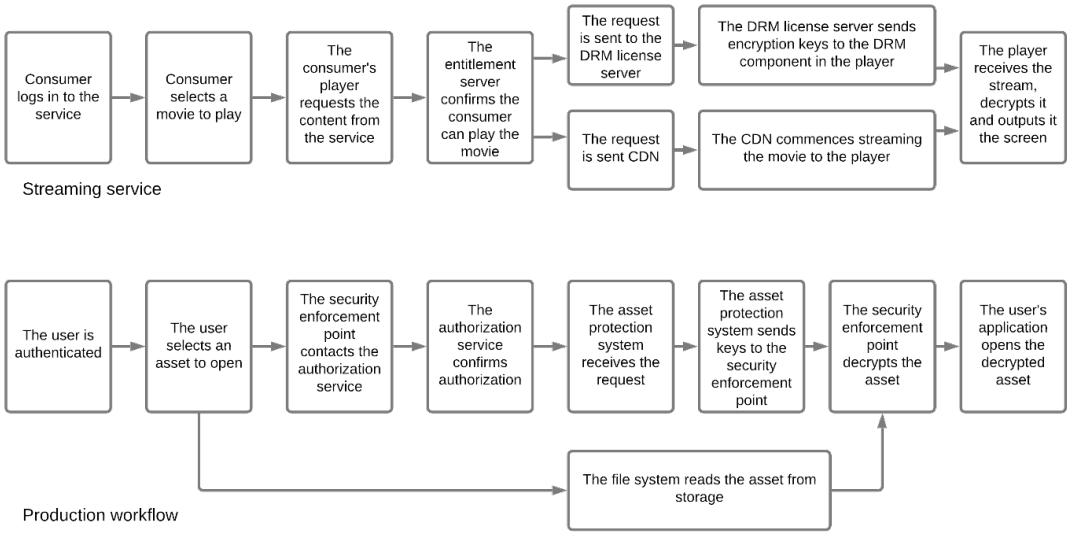

Figure 5-6 Asset encryption in streaming service and in production

We are not advocating that the solution is to use an actual DRM, but some useful parallels can be drawn.

[17] Of course, the assumption is that the encryption algorithm used is AES-128, AES-192, or AES-256 in the appropriate mode of operation. Currently, there are no known attacks on the AES algorithm where someone without knowledge of the key can read data encrypted with AES. The threat surface is insecure implementation and bad key management. See https://en.wikipedia.org/wiki/Advanced_Encryption_Standard#Known_attacks.

[18] Further information on NIST projects on roots of trust can be found using this link https://csrc.nist.gov/Topics/Security-and-Privacy/risk-management/roots-of-trust.

Asset encryption can be done without solution lock-in. If a common encryption algorithm operating in the same mode is used, for example AES-128 CTR mode, multiple solutions for content encryption can be used in parallel.

Consumer streaming services usually use more than one DRM because one DRM might be the only viable choice on one client platform but might not be available on another platform. For this reason, the majority of streaming services use more than one DRM. The same encrypted content is delivered to all players, but the key is delivered by the DRM that is supported by the target player. This is shown in Figure 5-7.

If asset encryption is used, it is not in the interests of the ecosystem to have a single technology provider, to require everyone working on a production to use the same technology, or to use only a technology that can be implemented on every platform.

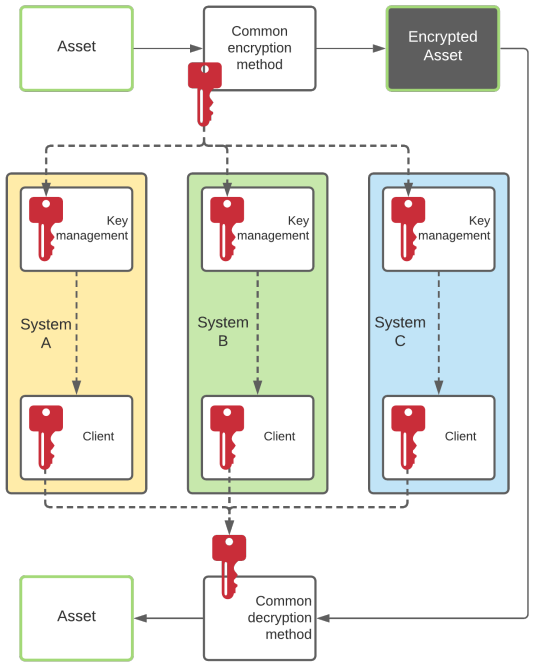

Figure 5-7 Asset encryption using multiple key distribution technologies

5.4 Policy Service

Up until now, we have described the components that come together to enable activity to take place. The authorization service is responsible for authorizing activity initiated by production management, subject to policy templates and the authentication service authenticating the entities involved.

The role of the policy service is to take the authorization policies created by the authorization service, process them, and turn them into actionable security that is acted on by the policy enforcement points.

The policy service has three components:

- Policy manager,
- Policy engine, and
- Policy enforcement point.

5.4.1 The Policy Manager

The policy manager is responsible for ensuring that authorized activity complies with global security policies and is secure in the current security situation, as reported by the continuous monitoring and security operations service and the threat analysis and intelligence service of the supporting security components.

The policy manager's role is to receive policies from the authorization service, reconcile them against global policies, then assess them against the current security situation. The precedence of global policies is a configuration option. If the policy manager finds no reason to prohibit the activity, the authorization

policy is sent to the policy engine. If the policy manager rejects a policy created by the authorization service, it sends an error message, which is then reported back by the authorization service to production management.

While global security policies could be incorporated into the templates used by the authorization service, the organization that generates the templates (e.g., the production) may not be the organization that sets global policies (e.g., the studio). Handling global policies in the policy manager avoids the need to ensure the templates are in sync with the global policies.

An example of a global policy would be a policy that prohibited the use of a particular vendor. If the authorization service generated a policy that involved that vendor, the policy manager would reject it.

The second function performed by the policy manager is the adjustment of policies in the light of the current security situation. The policy service is alerted to the current state of security by the continuous monitoring and security operations service and the threat analysis and intelligence service, both of which are part of the supporting security components.

An example of a change in the current security situation would be the discovery of a security vulnerability in a version of an application. If the authorization service generated a policy that authorized work using that application, the policy manager would add a restriction to that policy preventing the work from occurring on the vulnerable version of the application.

Another example of a change in the current security situation would be the determination that a user's account credentials had been compromised. The policy manager would reject policies that authorized that user to take part in an activity.

5.4.2 The Policy Engine

The policy engine stores and acts on the policies provided by the policy manager. Its role is to direct the policy enforcement points (PEP) described in Section 5.4.3 Policy Enforcement Points (PEPs) in real-time and in accordance with the policies passed to it. Ideally, the policies can be cached at the PEP, and cached ahead of work being done.

While it is advantageous from the point of view of minimum impact on workflows to generate authorization policies ahead of the initiation of an activity, consideration should be given as to how far ahead of time the policy engine loads policies into the PEP. Ideally, that would be just in time.

5.4.3 Policy Enforcement Points (PEPs)

Up until now, we have been describing the security system control plane. Now we are at the security system data plane.

The PEPs are embedded in the network, the devices, the applications, the storage system, the workflow management service bus, and so on. They enable authorized, and only authorized, activity to take place.

PEPs will vary according to the artifact they are protecting.

Earlier we mentioned that implementation of the architecture would benefit from a common policy description language, and the use of such a language would foster maximum interoperability in the PEPs.

5.4.4 Exception Handling in the Policy Service

There will be times when a policy enforcement point blocks an action, because the action is either:

1. An attempt to do something unauthorized, or
2. An attempt to do something that should have been authorized but has not been.

However, the PEP does not know which it is. All it knows is that something was attempted that is not allowed, causing it to send an exception to the policy engine.

The policy engine that receives the exception is able to do one of two things:

1. Send a new policy to the PEP, which will happen when the policy engine has a policy that allows the action, but the policy has not yet been sent to the PEP, or
2. Pass the exception up to its policy manager.

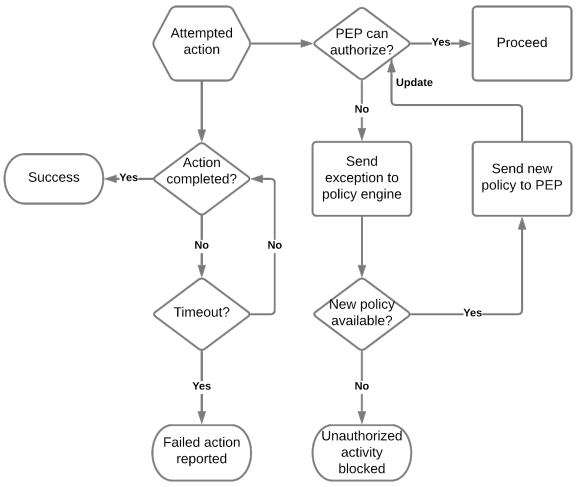

Figure 5-8 PEP exception process

The policy manager receives the exception only if there is nothing in the policy service that allows the action to take place. Depending on how workflows are managed, it may be possible for the policy manager to notify the appropriate workflow manager of the exception. If the workflow managers and the PEPs are on a service bus, the policy manager may have the information needed to notify the workflow manager and provide an identifier for the message.

Otherwise, in a real-world production with multiple workflow managers, the policy manager may not be able to determine which one to notify.

However, unless a new policy is sent to the PEP and the action allowed, the attempted action will fail and whatever initiated the action will become aware of the problem.

As with systems today, workflow management and asset management must be trusted – a subverted workflow management system can initiate activity that should not be permitted. The core components can validate the source of information sent from the workflow management and asset management systems, but unless the resultant policies created by the authorization service violate global policies, the core components cannot determine whether an activity should be authorized or should not. This can happen with existing security systems, possibly with fewer safeguards.

5.5 Distributed Security

This architecture supports the distribution of its components. There is, rightly, no single source of failure built into the architecture (although there may be in a particular implementation). There are many ways that the security can be distributed. Here are some examples:

1. A production has more than one workflow manager and an authorization service is associated with each workflow manager.
2. Authorization is delegated; for example, authorization for some part of a production is delegated to a vendor responsible for that work.
3. Separate security systems are deployed for each department involved in a production and authorization is hierarchical: the top tier authorizes the department to conduct a workflow, a second tier authorizes each task carried out by the department.
4. Policy services are specific to one class of authorization policy. For example, one policy service handles policies to do with processing and the other handles policies to do with asset access.
5. Services are replicated as a means of adding resilience, scaling capacity, and reducing latency by putting services close to where tasks are conducted.

There is nothing in the architecture that precludes any of these examples, or indeed distribution in general, but of course additional design work will be needed to define how this will work.

5.6 Summary of Authorization and Policy

The authorization service creates authorization policies by applying security policy templates to access permissions and creates authorization policies by applying security policy templates to activities created by the workflow management system.

The policy manager combines authorization policies generated by the authorization service with global security policies in the context of the security data from the continuous monitoring and security operations service and the threat analysis and intelligence service. The result is passed to the policy engine which controls the policy enforcement points.

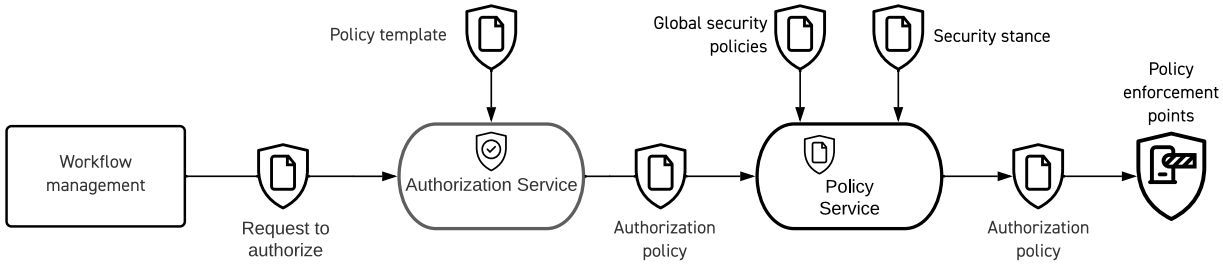

Figure 5-9 Policy creation flow

6 Supporting Security Components

Supporting security components are services used by the core security components to fulfill their roles.

This section describes the functionality of the supporting security components and the interaction with core security components.

The architecture defines the interfaces between the individual core component services using the six function components in Figure 6-1, but it does not define anything about the internal architecture or implementation of the supporting security components.

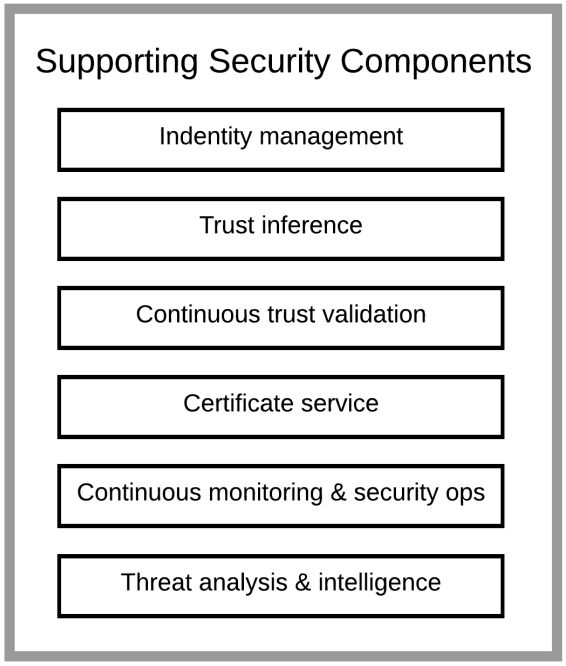

Figure 6-1 Detailed view of supporting components

6.1 Identity Management

Identity is at the core of any zero-trust architecture and of this security architecture. A robust way of verifying that a user, device, or application is what they claim to be is essential.

The identity management service would ideally be a single sign-on (SSO) system; however, it is not a requirement as long as the authentication service knows where to verify an identity.

The identity management service may be a federation of identity management systems, which is a group of mutually trusted identity management systems.[19]

Principle 6 in the 2030 Vision Paper states “Every individual on a project is identified and verified and their access permissions efficiently and consistently managed.” A MovieLabs work item arising out of this is work on a Production User ID (PUID), one function of which is to provide an identity management service for users with a PUID. That service would be part of an SSO federation.

6.1.1 Access Permissions

IAM systems provide both identity management and management of user access privileges. Here we distinguish between the two functions. The identity management part of IAM is used by the authentication service, and the access management part can be used by the authorization service. Figure 6-2 illustrates this point.

[19] https://en.wikipedia.org/wiki/Federated_identity

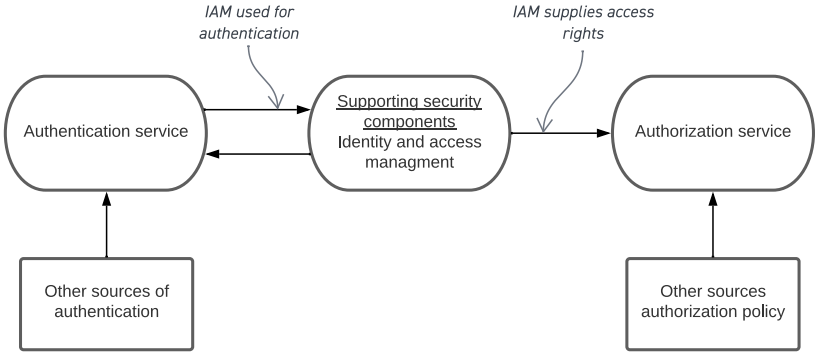

Figure 6-2 Relationship between an IAM and authentication and authorization

6.2 Trust Inference

Trust inference is the use of intelligence on user and device behavior to infer the level of trust that can be assigned at the time of an authentication request. It is also referred to as risk-based authentication.

Risk-based authentication is a dynamic component of authentication systems that considers parameters other than the credentials presented to determine how to perform authentication.

For example, if a website frequently presents a captcha challenge when you attempt to log in, that may be a result of a trust assessment that has detected something out of the ordinary, such as an IP address with a different locale from your usual log in location.

The level of trust that can be inferred may be used to increase or decrease the complexity of the authentication mechanism. If the trust inference yields a high score, user authentication may happen with a simpler PIN. For example, the PIN on a Windows 10 workstation is not as complex as a password because trust is inferred by the user typing on a physical keyboard attached to the computer.

When trust inference yields a low score, additional measures could be required, up to and including identifying the authentication request as an attempted intrusion and denying the request regardless of credentials presented.

This has two benefits for the production. In addition to increasing the security of the authentication, it lightens the touch of the user authentication process, often a point of friction for the user.

Denying an authentication request based on trust inference need not affect legitimate activity. By differentiating between a possible attack and a legitimate log in attempt, user activity that scores well on trust inference is unimpeded, whereas activity that scores low in trust inference can be blocked.

Denying an authentication attempt because it has a low trust score is not the same as locking the user account after a certain number of failed password attempts. That method of blocking an attempted intrusion also blocks legitimate access. Trust inference could block login attempts with a particular set of parameters (e.g., geographic location), while not interfering with other login attempts for the same account with a different set of normally present parameters.

Of course, levels are set as a matter of policy by the organization and do not have to be uniform for all users and devices.

A common method of trust inference is contextual or risk-based authentication management. An example is:

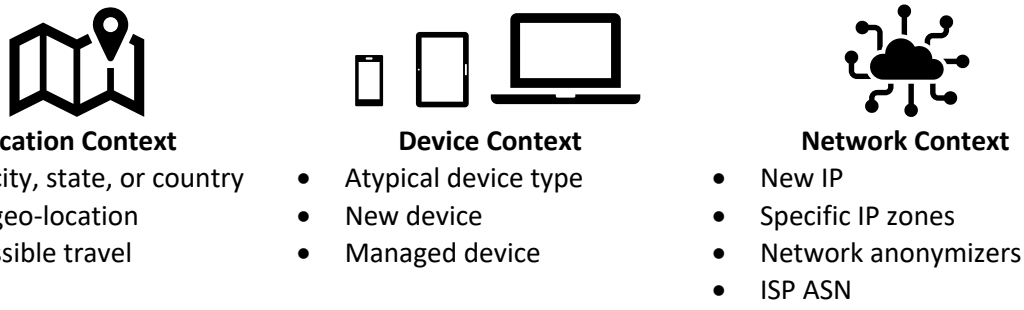

Figure 6-3 Contextual access management (source: Okta)

Trust inference can also be used to deal with devices that are subverted, have known vulnerabilities, or are not managed by the production. These devices may be treated differently (including denial of all connections) than devices owned or registered by the production that are deemed in a secure state.

Both trust inference and continuous trust validation use behavioral attributes, including automated subject analytics, device analytics, and measured deviations from observed usage patterns.

By using trust inference and continuous trust validation (Section 6.3), the authentication service is able to reduce the burden on users to present credentials when they log into a new system, while also increasing security by requiring re-authentication or denying access when artifacts engage in unexpected behavior.

6.2.1 Trust Learning

Production is a fluid environment. Processes change during a production and from one production to another. Production is carried out by a substantial number of individuals contracting directly with the production or employed by any number of other organizations such as the studio, post houses and VFX companies.

The operation of trust inference and the associated continuous trust validation, described in Section 6.3, uses knowledge of the expected behavior that can be learned from the activities over a period of time, correlating activity with authorized activity, and so on.

6.3 Continuous Trust Validation

In the previous section we described trust inference, i.e., the use of activity intelligence in the decision to establish trust.

Continuous trust evaluation is the process of determining if an established trust relationship should still be trusted.

Question	Mechanism
Should I trust this entity?	Identity management and trust inference
Should I still trust this entity?	Continuous trust evaluation

The same, or similar, parameters used in trust inference are collected from real-time activity intelligence, including data acquired by network and system logging. Analysis of that activity determines whether a trusted entity should still be trusted.

A simple example is a situation where trust is established for a user connecting from one IP address and in mid-session the IP address changes to a different one in a different region. A circumstance like this might happen when an attacker is using a mechanism to disguise location such as Tor (anonymity network[20]) or a VPN. The change in the IP address triggers an "impossible travel" rule[21] and the trust is revoked.

When continual trust evaluation detects a potential issue, it does not always have to result in the immediate termination of user activity. It may trigger reauthentication or increased surveillance.

When the trust score created by trust inference is at the lower end of the acceptable range, it may be that the continuous trust validation uses tighter constraints so that even a slight negative change in the trust assessment alerts the authentication service to take action, such as re-establishing trust through re-authentication.

6.4 Certificate Service

The certificate service is the certificate authority responsible for generating and tracking certificates issued to users, systems/devices, applications, and the asset protection service, as well as managing and distributing Certificate Revocation Lists (CRL).

The certificates are used for cryptographic purposes throughout the system.

All certificates are revocable and system components using the certificates should check current validity and not rely solely on expiration times.

6.5 Continuous Monitoring and Security Operations

Continuous monitoring and security operations (CMSO) is a set of functions conducting real-time analysis of multiple data sources to provide situational awareness to other security components and to the information security operations center (ISOC).

The functions that fall under the header of continuous monitoring and security operations include functions variously known as:

- Network security monitoring

[20] https://en.wikipedia.org/wiki/Tor_(anonymity_network)

[21] For an explanation of impossible travel and how the Microsoft Cloud App Security addresses it, see https://www.daymarksi.com/information-technology-navigator-blog/understanding-office-365-impossible-travel.

- Continuous security monitoring
- Continuous diagnostics and mitigation

Together these functions:

- Monitor system and network activity
  - The collection and analysis of data about network traffic, access requests, processes, access requests, locations, and so on.
- Monitor user actions
  - The collection and analysis of data about user actions, including authorized activity, attempts at unauthorized activity, user location, and so on.
- Monitor assets
  - The collection and analysis of data about access to assets, movement of assets, network traffic associated with access to assets, location of assets

These functions provide the security architecture with notification of security events, including attempted intrusions and devices falling out of compliance with security requirements.

The data collected may also be used for provisioning cloud resources and networks, as well as generating alerts for system performance problems.

6.6 Threat Analysis and Intelligence

The functions under the heading of threat analysis and intelligence provide analysis of the threat environment based on the collection of data from within the security system and outside of it. The emphasis here is on the depth of analysis rather than real-time sensors.

- Security information and event management (SEIM) collects security-related data for later analysis. This data is used to refine underlying policies and warn of possible attacks.
- Threat intelligence collects information from internal and external sources that helps in the management of authentication and authorization policies.
- Activity logs are collected from devices, network components and security enforcement points.

7 Production Management

The purpose of the security system is to enable secure production. Consequently, the core security components are directed by production management to authorize work.

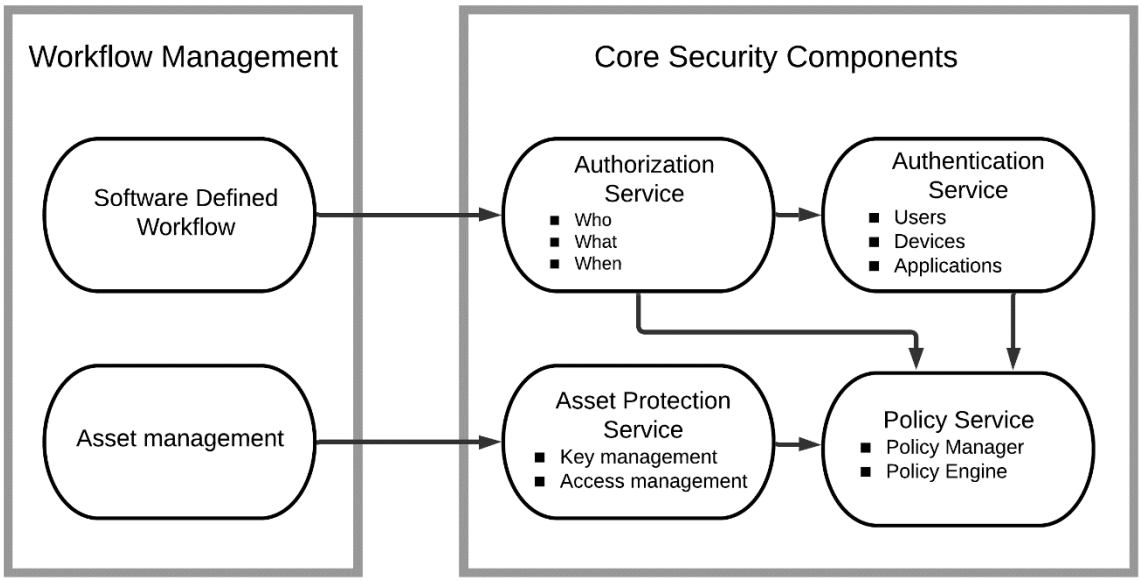

Figure 7-1 Interaction with production management

The workflow management system notifies the authentication service of the participants, devices, application services and applications that are part of the workflow so that the authentication service can ensure proper authentication. The notification might be applicable to a group of participants or devices, or a range of versions of an application.

The workflow management system sends a complete description of the workflow (or a part of it) to the authorization service. That description includes:

- A list of the participants, devices, application services and applications
- A list of tasks along with parameters such as duration of the task

The asset management service sends a list in the form of asset identifiers and locators of the assets that are required for the workflow.

A timeline of the interaction between production management, the core security components, and the supporting security components looks something like this:

Figure 7-2 Flow of workflow authorization

The exact flow of workflow authorization depends on how workflow is managed. For example, if workflow is managed by a traditional scheduling system that creates work orders, operation is different than if the workflow is managed by an orchestration system.

8 Wrap-up

This architecture is, like any architecture, a framework for building implementations. It is defined using two groupings. The first group is the set of core security components that secure production workflows. The second group is the set of supporting security components that are used by the core components. These components are available from the hyper-scale cloud providers and security-as-a-service vendors.

Our descriptions of the supporting security components are not intended to imply how they should be implemented or divided up. The interfaces between the core and supporting security components are interfaces between the core security components and a black box that contains services that provide the requisite functionality.

The core security components connect to the workflow management system since it is the system that determines what activities should take place and therefore what should be authorized. The security architecture does not, and should not, define how the production management system operates. The only requirement is that production management use the defined interfaces.

With that in mind, we can redraw the high-level architecture diagram presented earlier, Figure 4-1, in this way:

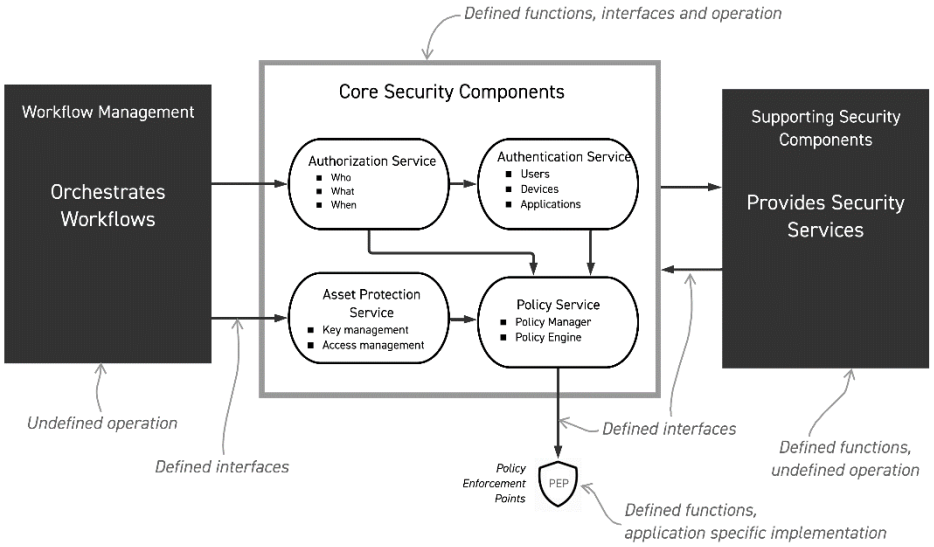

Figure 8-1 High-level architecture re-expressed

Even though there are complex interactions happening to secure a production, it is important to note that most of the services are never exposed to creatives involved in a production workflow. If the workflow, tools, and systems are designed to be secure, the users should rarely, if ever, interact directly with the components of the security system.

One note about Figure 8-1: the word “interface” is used in a broad sense and does not necessarily mean an API.

8.1 Next Steps

This description of the architecture is just a beginning. We expect the architecture to evolve as the industry works together to facilitate implementations. Notable next steps include defining interfaces and developing a policy description language.

8.1.1 Interfaces

The security of the production ecosystem is best served if the security architecture can be implemented to secure production activity regardless of the cloud service used, whether a hyper-scale provider or a private cloud, and regardless of whether more than one cloud is used. Implementation using common interfaces serves that goal and also provides interoperability, which makes it easier to reconfigure security systems using different components. Many of the interfaces between components can use existing APIs. But the architecture does include components that will require the development of new interfaces.

An important next step on the road to implementation will be determining how to leverage existing standardized or commonly used interfaces for communication between components. Where new interfaces important to production security are required and not already under development, our industry may need to work together to develop those new interfaces.

Possible areas for interface development include:

1. Interfaces between the core security components and the supporting security components. Part of this work could include further defining the functions performed by the supporting security components.
2. Interfaces between the core security components and production management. It should be anticipated that more than one set of interfaces may be needed to accommodate approaches to production management that are significantly different.
3. Interfaces between the policy service and the policy enforcement points. Part of this work could be defining the functions performed by each type of PEP (e.g., a PEP controlling access to assets, a PEP controlling the use of an application, etc.)
4. Interfaces between the core components, including development of functional specifications for the core components.

Depending on the level of interoperability required between specific components, interface development could range from best practices and conventions to fully specified APIs.

8.1.2 Policy Description Language

This architecture is built around authorizations which are manifest in the policies that propagate to the PEPs, the endpoints of the system. This is where the architecture is potentially most complex.

Taking advantage of the full capability of the architecture requires normative descriptions of policies that are actionable and unambiguous. An obvious next step is therefore the definition of a policy description language.

There is also the issue of how workflow managers inform authorization services of activity to be authorized. That issue could be addressed as part of the policy description language or as an outcome of work on standardizing software-defined workflows.