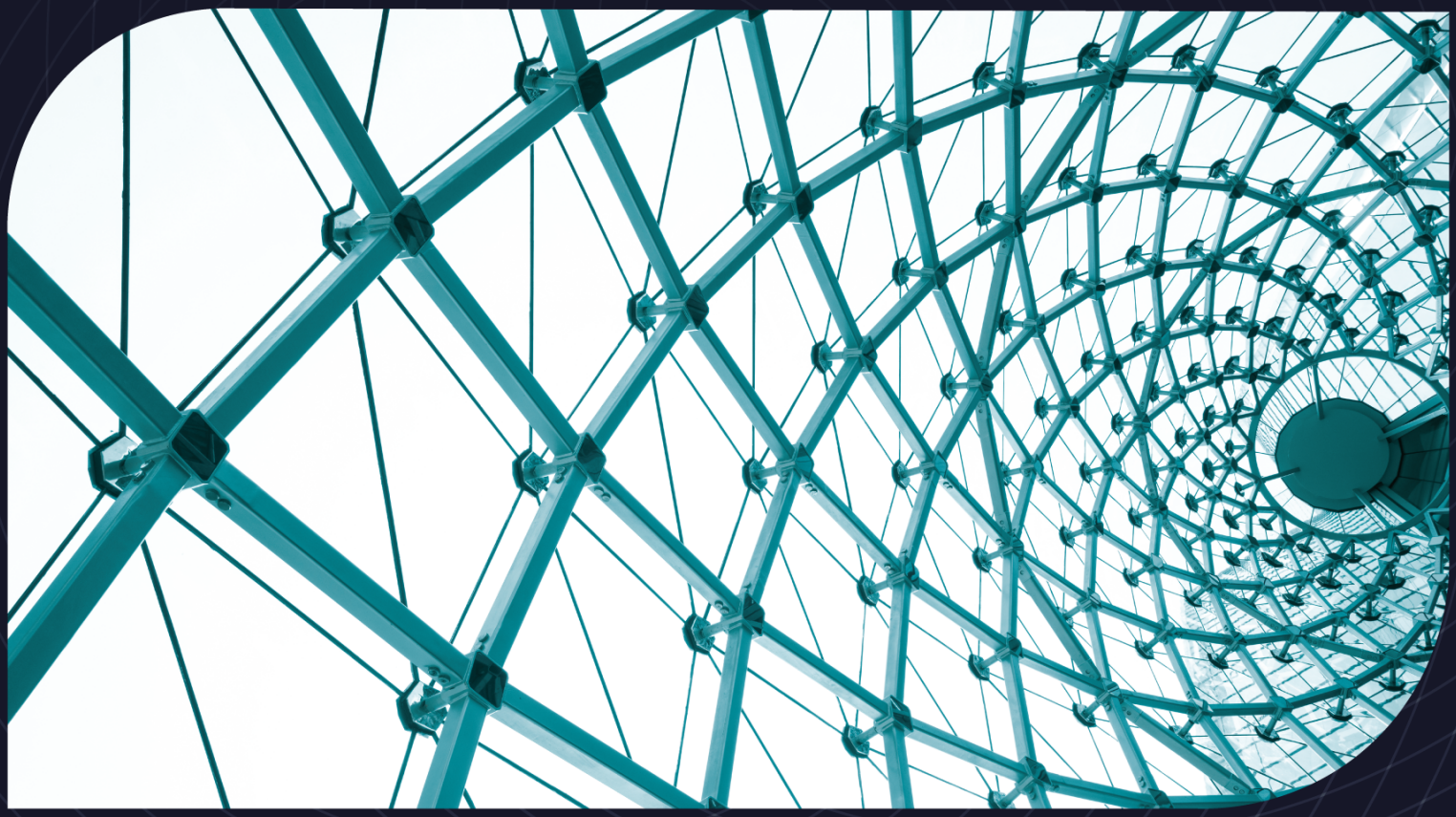
Architectures & Components of Software-Defined Perimeter

CCZT Study Guide



cloud security alliance®

CCZT™
Certificate of
Competence in Zero Trust

The official location for SDP and Zero Trust Working Group is
https://cloudsecurityalliance.org/research/working-groups/zero-trust/

Disclaimer

Cloud Security Alliance designed and created this Zero Trust Training course study guide (the "Work") primarily as an educational resource for security and governance professionals. Cloud Security Alliance makes no claim that use of any of the Work will assure a successful outcome. The Work should not be considered inclusive of all proper information, procedures and tests or exclusive of other information, procedures and tests that are reasonably directed to obtaining the same results. In determining the propriety of any specific information, procedure or test, professionals should apply their own professional judgment to the specific circumstances presented by the particular systems or information technology environment.

Version Number: 20240820

About Cloud Security Alliance

The Cloud Security Alliance℠ (CSA) (www.cloudsecurityalliance.org) is the world's leading organization dedicated to defining and raising awareness of best practices to help ensure a secure cloud computing environment. Cloud Security Alliance harnesses the subject matter expertise of industry practitioners, associations, governments, and its corporate and individual members to offer cloud security-specific research, education, certification, events and products. Cloud Security Alliance activities, knowledge and extensive network benefit the entire community impacted by cloud—from providers and customers, to governments, entrepreneurs and the assurance industry—and provide a forum through which diverse parties can work together to create and maintain a trusted cloud ecosystem.

CSA Address

709 Dupont St.
Bellingham, WA 98225, USA
Phone: +1.360.746.2689
Fax: +1.206.832.3513

Contact us: support@cloudsecurityalliance.org
Website: https://cloudsecurityalliance.org/
Zero Trust Training Page: https://knowledge.cloudsecurityalliance.org/page/zero-trust-training
Zero Trust Advancement Center: https://cloudsecurityalliance.org/zt/
Provide Feedback: support@cloudsecurityalliance.org
CSA Circle Online Community: https://circle.cloudsecurityalliance.org/
Twitter: https://twitter.com/cloudsa
LinkedIn: www.linkedin.com/company/cloud/security/alliance
Facebook: www.facebook.com/csacloudfiles
CSA CloudBytes Channel: http://www.csacloudbytes.com/
CSA Research Channel: https://www.brighttalk.com/channel/16947/
CSA Youtube Channel: https://csaurl.org/youtube
CSA Blog: https://cloudsecurityalliance.org/blog/

Acknowledgments

Dedicated to Juanita Koilpillai, a pioneer in software-defined perimeters whose contributions to the Certificate of Competence in Zero Trust (CCZT), Zero Trust training, and CSA are immeasurable.

The CCZT and Zero Trust training was developed with the support of the Cloud Security Alliance Zero Trust Expert Group, whose members include volunteers from a wide variety of industries across the globe. Made up of subject matter experts with hands-on experience planning and implementing Zero Trust, both as cloud service consumers and providers, the Zero Trust Expert Group includes board members, the technical C-suite, as well as privacy, legal, internal audit, procurement, IT, security and development teams. From cumulative stakeholder input, the Zero Trust Expert Group established the value proposition, scope, learning objectives, and curriculum of the CCZT and Zero Trust training.

To learn more about the CCZT and Zero Trust training and ways to get involved please visit: https://cloudsecurityalliance.org/education/cczt

We would also like to thank our beta testers, who provided valuable feedback on the CCZT and Zero Trust training: https://cloudsecurityalliance.org/contributors/cczt-contributors

Lead Developers:

Aunudrei Oliver
Heinrich Smit
Michael J. Herndon
Michael Roza
Prasad T.
Richard Lee
Robert Morris
Shruti Kulkarni
Vani Murthy

Contributing Editors:

Alex Sharpe
Cory Missimore
CT Chidam
Mark Mcgloin
Matt Lee

Expert Reviewer:

Clément Betacorne
Ledy M.
Peter van Eijk
Roland Kissoon

CSA Global Staff

Anna Schorr
Chandler Curran
Daniele Catteddu
Hannah Rock
Ian Erwert
Leon Yen
Noelle Sheck
Stephen Smith

Table of Contents

List of Figures

Course Introduction

Welcome to Architectures & Components of Software-Defined Perimeter by the Cloud Security Alliance (CSA). This training module is part of a larger series of CSA courses focused on Zero Trust (ZT) and Zero Trust Architecture (ZTA). CSA's Software-Defined Perimeter (SDP) provides organizations with a flexible, vendor-agnostic approach to protecting IT infrastructures from increasingly sophisticated cyber threats. Previous courses in this series provided learners with an introduction to ZTA and SDP, as well as a comprehensive overview of SDP's key features and technologies. In this course, learners will get an in-depth look at SDP's main components, workflows, communication flows, and logging, as well as SDP and NIST ZTA deployment models.

Course Structure

This course consists of five units, each geared toward helping learners gain competency in the following topics:

- SDP Components
- SDP Workflows
- SDP Communication Flows
- SDP Logging
- SDP and NIST Deployment Models

Course Learning Objectives

After completing this course, learners will be able to:

- Understand the main components that make up an SDP architecture
- Describe the onboarding and access workflows established for supporting SDP processes
- Describe the communication flows that occur between SDP components
- Explain the SDP logging mechanisms and the role they play in SDP architectures
- Understand CSA's SDP and NIST's ZTA deployment models

1 SDP Components

To achieve an optimal level of ZT, organizations should eliminate implicit trust models and adopt more dynamic, granular methods of securing access and interactions to protected resources. SDP's architectural components work in concert to provide dynamic security controls for addressing the challenges of traditional network infrastructures. Together, the SDP components interact through a context-based, policy-driven framework—one that focuses on securing critical organizational resources versus protecting the physical network perimeter.

SDP's foundational components are introduced in CSA's Introduction to Software-Defined Perimeter and are covered in the following resources:

- Software-Defined Perimeter[1]
- SDP Specification v2[2]
- SDP Architecture Guide v2[3]

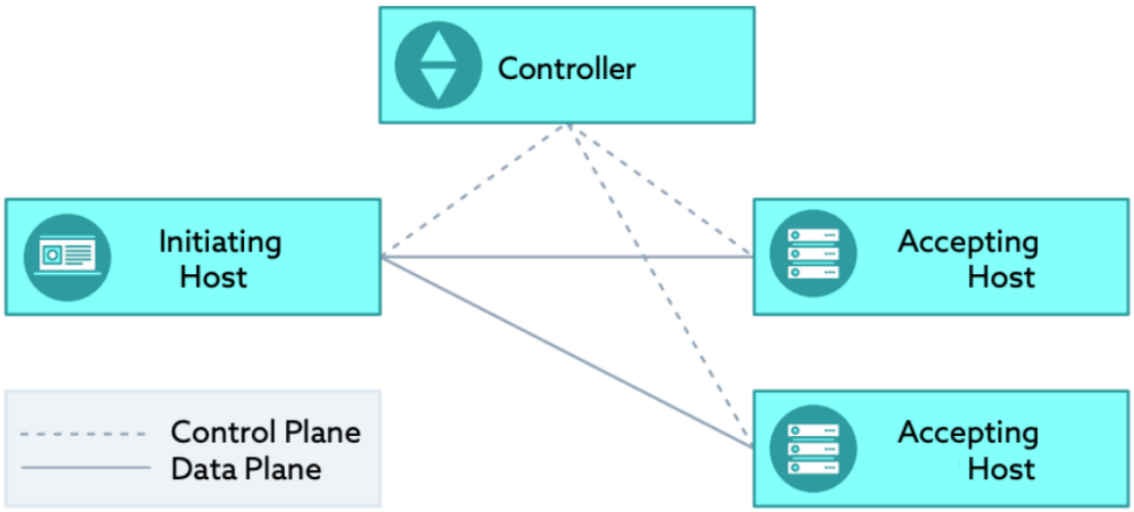


Figure 1: *SDP Components and Interactions*

The following sections delve into each one of the key SDP architecture components:

- Initiating host (IH)
- Controller
- Accepting host (AH)
- Gateway/resource

1.1 Initiating Host

The IH commonly consists of an agent running on an accessing entity—this could be an endpoint device (e.g., laptop, tablet, smartphone) or a non-person entity (NPE) such as a server, network device, web browser, software application, or service. Software application IHs provide richer capabilities such as host checking (i.e., device posture checks), traffic routing, and more streamlined authentication measures. Crucially, the IH initiates connections using single packet authorization (SPA), with SPA packets generated by browser-based SDP clients in some implementations. IHs communicate with the controller in order to access the organization's resources using the AHs; during authentication, IHs may provide the controller with information like the user's identity, hardware/software inventory of the endpoint device, and the endpoint device's health status. Upon access authorization, the controller must also provide a mechanism for the IH to communicate with the AH securely.

[1] https://cloudsecurityalliance.org/artifacts/software-defined-perimeter/
[2] https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-zero-trust-specification-v2/
[3] https://cloudsecurityalliance.org/artifacts/sdp-architecture-guide-v2/

1.2 Controller

The controller acts as a policy definition, verification, and decision mechanism that maintains information about which identities (e.g., users, groups) from which devices should be granted access to an organization's resources. Also referred to as the policy decision point (PDP) in NIST's ZTA model, the controller uses SPA to render itself and other SDP components invisible and inaccessible to unauthorized users/devices. This mechanism may be provided by an SDP gateway fronting a controller, or natively within the controller itself.

Chiefly, the controller is responsible for determining which SDP components can communicate with each other. When an IH initiates communications with the controller to access a specific resource or service, the controller grants access via the AH based on the IH users' privileges. To perform IH user authentication, the controller may use an internal user table or connect to an identity provider (IdP), as well as enforce multi-factor authentication (MFA).

The following figure depicts a workstation/IH inside a corporate network with an AH co-resident/protecting the resource, an enterprise laptop with enterprise identity access management (IAM), and installed IdP agents.

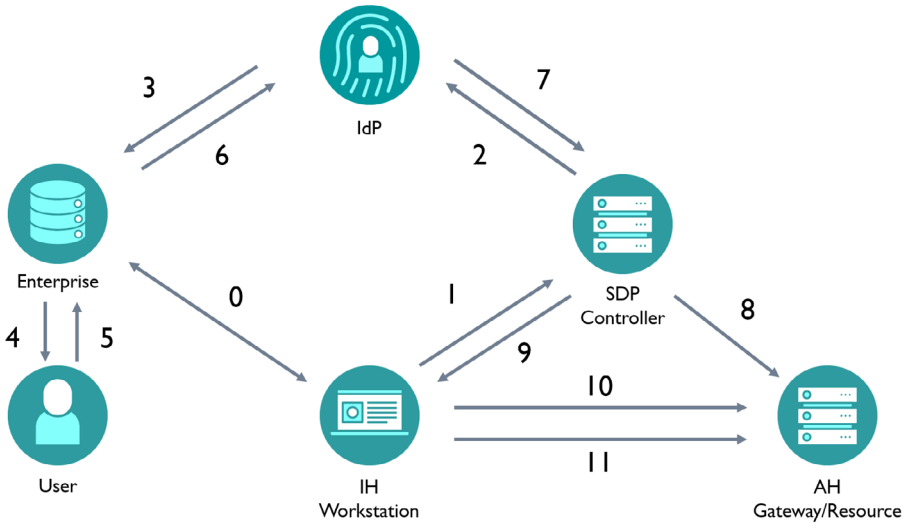


Figure 2: *Enterprise Laptop/IH Inside a Corporate Network*

In this scenario, an SDP-onboarded workstation/IH is requesting access to the gateway/resource. The interactions between the SDP controller and components depicted in this figure are as follows:

0. The **user/workstation** authenticates to the appropriate **enterprise** services (assuming the workstation has already been onboarded)
1. The **IH** sends an SPA packet to the **SDP controller**
2. The **SDP controller** initiates a request to the **IdP**
3. The **IdP** sends a request to the **enterprise**
4. The **enterprise** IAM initiates MFA for the **user**
5. The **user** responds to the **enterprise** MFA request
6. The **enterprise** responds to the **IdP**

7. The **IdP** responds to the **SDP controller**
8. The **SDP controller** sends information to the **AH**
9. The **SDP controller** responds to the **IH**
10. The **IH** sends an SPA packet to the **AH** (assuming the controller has approved the connection)
11. The **workstation** initiates an mTLS connection to the **gateway/resource**

In this example, the AH's role is equivalent to that of a NIST ZT model's policy enforcement point (PEP), while the controller performs the functions of a NIST ZT model's PDP.

1.3 Accepting Host

The AH is a logical SDP component that fronts applications, services, and resources accessed and protected by the SDP. Typically residing on a network under the enterprise's and/or a direct representative's control, AHs usually consist of an agent or host that utilizes the SDP control plane to broker IH access to protected resources. AHs accept connections from IHs, terminate the mTLS sessions initiated by the IH in the data plane, and—based on instructions from the controller—either deny access or provide an encrypted tunnel for data transfer and access to protected resources/services.

As illustrated in the following diagram, the AH may be co-resident with the target service or separated by a network.

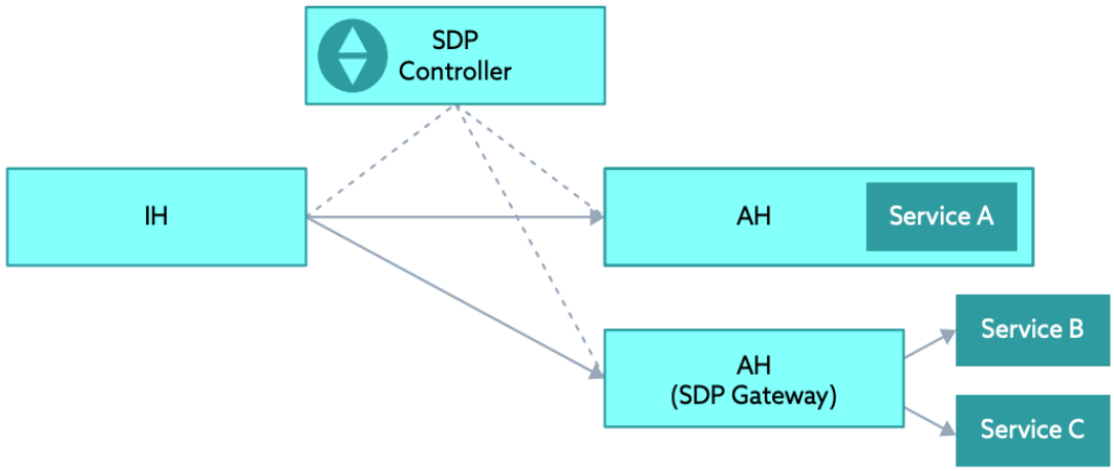


Figure 3: *AH and SDP Protected Resource Configurations (Co-Resident or Separated)*

A crucial AH feature is the drop-all firewall—using this security mechanism, the AH implicitly denies all connections and drops all unsolicited packets. When the device and user-level authentication is successful, the IH forwards the request to the controller for further authentication (e.g., authentication with a third-party IAM/IdP or MFA provider) and session establishment. If the request is approved, the connection between the AH and IH is established.

1.4 Gateway

A gateway is employed if one or more servers require isolation and stronger access controls for their protected services. Gateways ensure that only authorized users and devices can access protected resources and that all other traffic is dropped. Additionally, monitoring, logging, and reporting capabilities can be implemented on gateways for more granular details behind these connections.

The following diagram depicts two possible gateway configurations in different SDP deployment model architectures, with gateway/AH traffic highlighted in red. The gateway/AH establishes virtual boundaries around internet-connected assets and user activity. Depending on the SDP deployment model, the gateway may be co-located on the AH or deployed in front of it.

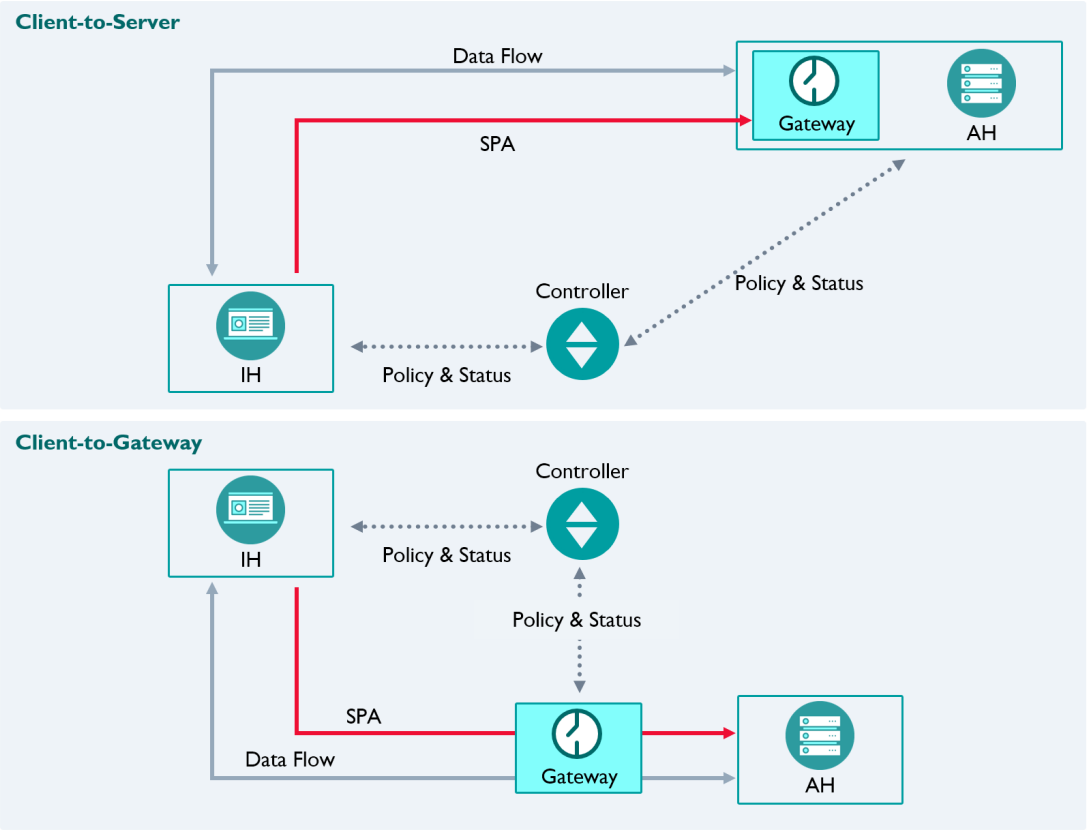


Figure 4: *Logical Gateway Configurations—Client-to-Server vs. Client-to-Gateway*

In the case of client-to-server SDP deployments, the gateway is co-located on the AH. In contrast, client-to-gateway SDP deployments place the gateway in front of the AH for enhanced service/resource obfuscation from the external environment. Other SDP deployment models may incorporate additional gateway configuration variations—for example; the gateway-to-gateway model has the gateway logically serving as both the IH and AH. SDP deployment models are covered more in-depth later in this course.

2 SDP Workflows

Broadly speaking, SDP component workflows can be grouped into two categories:

- Onboarding Workflows: for initializing SDP components prior to communicating, with separate flows for each component
- Access Workflows: for supporting continuous SDP interactions, coordinated among multiple components

Different SDP implementations and deployment models may vary in terms of specific interactions. In general, each SDP component typically has a single onboarding workflow, followed by participation in multiple access workflows.

2.1 Onboarding Workflows

An SDP component requires initial onboarding before communicating with other components. This section describes the onboarding, authentication, and initial communication between the IH, AH and controller, including the following workflows:

- Controller Onboarding Workflow
- AH Onboarding Workflow
- IH Onboarding Workflow
- Onboarding Authentication & Authorization

2.1.1 Controller Onboarding Workflow

An SDP architecture requires one or more controllers, with at least one continuously available to support any pending and/or active onboarding workflows. Some implementations also require an active controller to support pending and/or active access workflows. Controllers must be network-accessible from all locations where SDP components reside and operate (i.e., globally accessible from the internet, but only for authorized users and devices).

The following diagram depicts the workflow of an initial/primary controller brought into service and connected to the appropriate authentication and authorization services (e.g., PKI and certificate authority service, IAM, MFA, device attestation service). When required, the controller runs continuously and indefinitely to provide immediate, on-demand support to other SDP components.

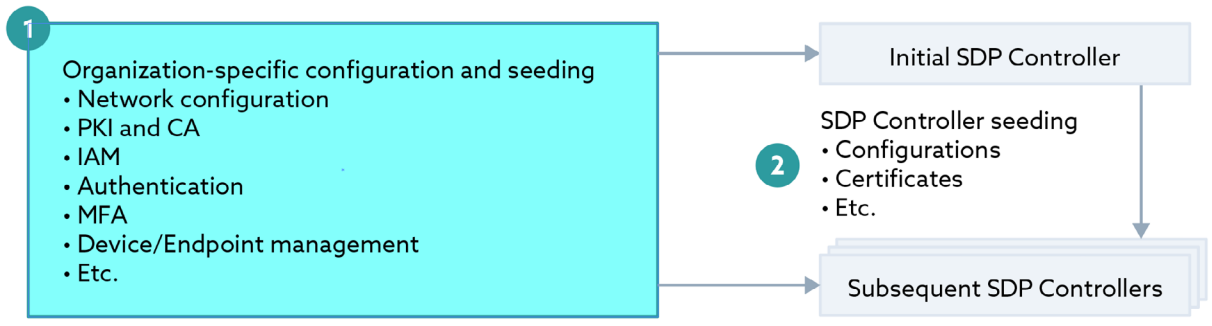


Figure 5: *Controller Onboarding Workflow*

Optionally, subsequent controllers may be onboarded with a combination of the same organization-specific configurations, as well as specific configuration details from the initial controller.

SDP implementations may also include the deployment of a cluster of controllers for redundancy and high availability. In this case, each SDP implementation must support a mechanism by which subsequent controllers are connected to others in cluster formation, acting as a single logical node with the same shared state.

2.1.2 Accepting Host (AH) Onboarding Workflow

SDP architectures require the deployment of one or more AHs configured for the selected SDP deployment model. For example, AHs may be configured as standalone gateways or deployed as part of a server (i.e., co-resident with the protected resource/workload).

In the following AH onboarding workflow, the initialized AH must connect and authenticate to one or more controllers in the SDP. Once onboarded, the AH is ready to receive SPA packets and support requests from authorized IHs.

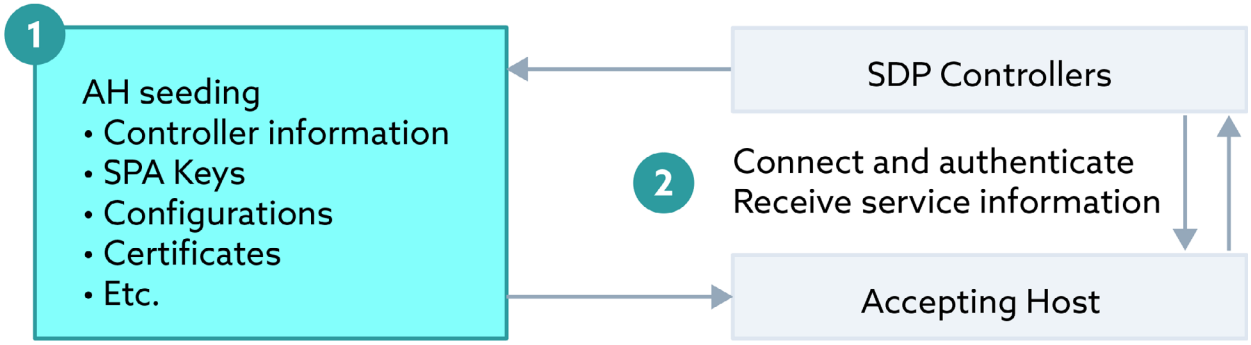


Figure 6: *AH Onboarding Workflow*

Like controllers, AHs can be configured as a cluster for high availability and redundancy. In these scenarios, the SDP implementations must support a mechanism by which AHs are configured to connect to the controllers in its cluster. SDP architectures with AH clusters typically use one of the gateway deployment models.

AHs may be long-lived or ephemeral, as both are acceptable in SDP implementations. Standalone gateway AHs may be long-lived (e.g., an AH system running for months/years) or short-lived (e.g., a dynamic gateway cluster that scales on-demand). When deployed within an individual server/workload, an AH may also be long-lived or short-lived—in this case, its lifespan is determined by the underlying server instance. For example, server instances may be long-lived (e.g., traditional web/application servers) or short-lived (e.g., part of an ephemeral DevOps infrastructure).

2.1.3 Initiating Host (IH) Onboarding Workflow

Like all SDP components, IHs require onboarding; during the onboarding workflow, IHs are configured with the initial information required for connecting to the controller. This includes network information (e.g., hostnames, IP addresses), as well as any required shared secrets (e.g., SPA keys, certificates). IHs only require onboarding once to initiate the access workflow.

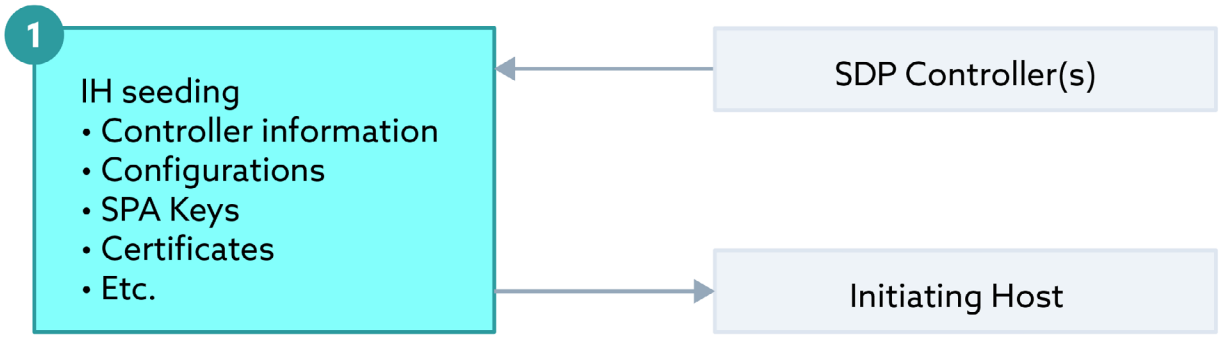


Figure 7: *IH Onboarding Workflow*

IH onboarding commonly incorporates an enterprise identity management provider and, for user devices, may incorporate user authentication as a step within the IH onboarding workflow. The following example illustrates an IH onboarding workflow that leverages an IdP for authentication.

2.1.3.1 IH Onboarding Workflow Example

In the following IH onboarding workflow, the IH authenticates with the controller to connect with the AH. In turn, the controller requires information such as the IH's identity, the IdP that authenticates the IH to the controller, and the AH that the IH will connect to. Controllers must be configured and onboarded before any gateways or IHs/AHs.

At a high level, IH onboarding requires the following information:

- The identity of the IH initiating the connection to the AH
- The IdP authenticating the IH with the controller
- The authentication factor(s) used by the IdP (e.g., certificates, keys, MFA)
- The AH the IH is allowed to connect to
- The context in which the IH is allowed to connect to the AH, as configured by the controller)

The following SDP onboarding figure, taken from the SDP Specification v2, illustrates one possible IH onboarding workflow. In this scenario, the IdP integrates with the controller, user, and device.

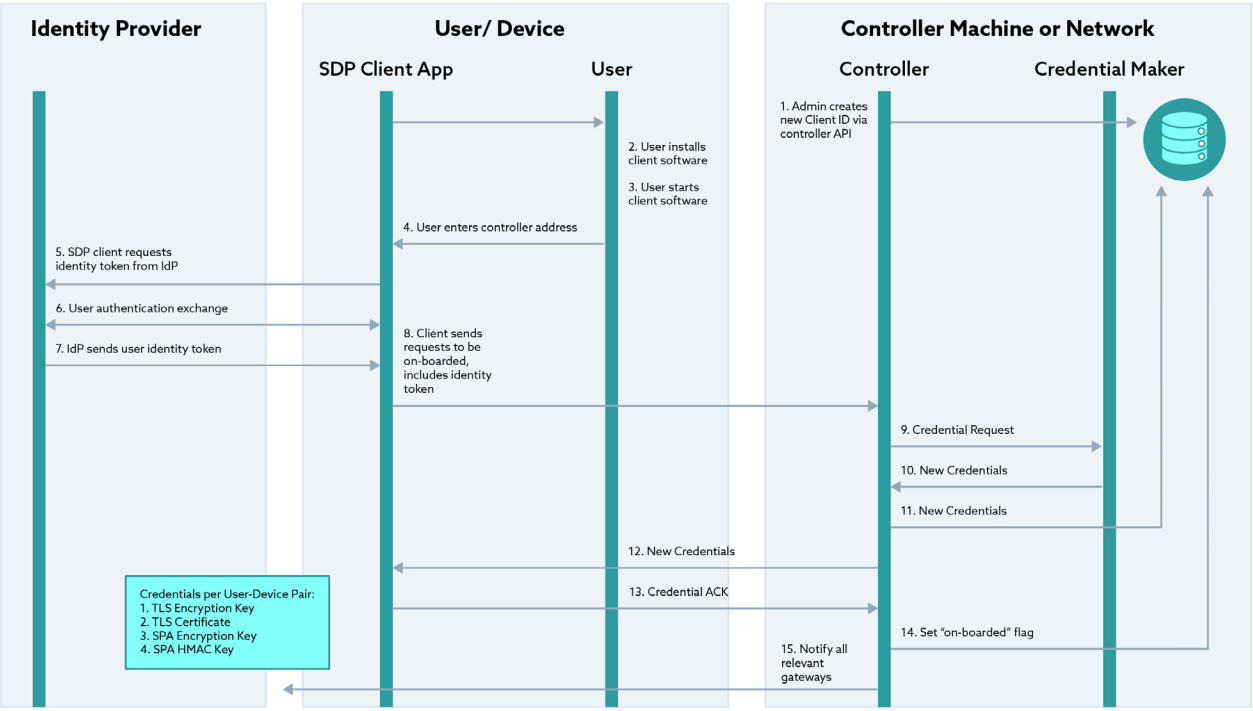


Figure 8: *SDP Onboarding (IdP Scenario)*

The example IH onboarding workflow with an IdP consists of the following steps:

1. The admin creates a new client ID via the controller API and inserts the record into the credential database.
2. The admin also creates a policy for the user's access to the applications. An existing policy also can be leveraged as per the situation.
3. Client software is installed on the IH using an administration solution/mechanism (e.g., GPO, MDM, part of a base image).
4. The user starts the client software and the user's device is configured to connect to the controller.
5. The client requests an identity token from the IdP.
6. The user authentication exchange process occurs.
7. The IdP sends the user identity token.
8. The client sends the onboarding request (including the identity token).
9. The credential request is sent.
10. New credentials are sent from the credential maker to the controller via a separate REST API.
11. New credentials are sent from the controller to the credential database.
12. New credentials are sent from the controller to the client app.
13. The SDP client app sends a credential acknowledgment (ACK) to the controller.
14. The controller sets the onboarded flag and updates the credential database.
15. The controller notifies all relevant gateways that the user/device has been onboarded.

In step 5 of the scenario depicted above, the IH uses an SDP client app to negotiate with the IdP. In some SDP deployment models, the controller will have a pre-established trust relationship with the IdP (e.g., to validate a SAML token forwarded by the client). Yet other scenarios may involve authentication managed by the controller.

2.1.4 Onboarding Authentication & Authorization

The following workflow is an example of the dynamic authentication and authorization of an IH to the AH, leveraging the controller's integration with the enterprise IAM/IdP system. In this scenario, a dynamic authorization check occurs involving an enterprise laptop requesting an SDP resource. The enterprise laptop has IAM, IH, and IdP agents installed, and the IH requests access to an authorized resource protected by an SDP gateway.

During the process of onboarding, the controller sends the authorized AH list to the IH. The controller may also send each AH a list of authenticated IHs.

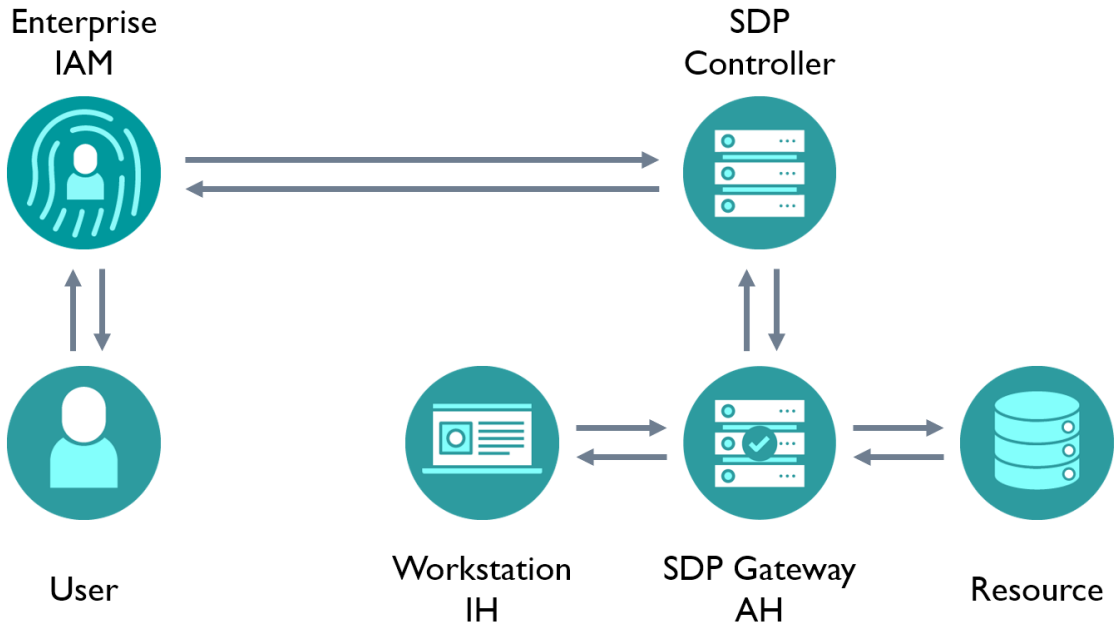


Figure 9: *Onboarding Authentication & Authorization*

Step	Phase	Description
1	Authorization	The IH sends an SPA packet to the AH/gateway requesting access to the resource
2	Authorization	The AH/gateway sends the access request to the controller
3	Authorization	The controller sends the access request to the enterprise IAM to verify that the AH is authorized
4	Authentication	The enterprise IAM initiates MFA to the enterprise user
5	Authentication	The user responds to MFA
6	Authentication	The enterprise IAM responds to the controller
7	Authentication	The controller responds to AH/gateway
8	Authorization	The AH/gateway sends the request to the resource
9	Authorization	The requested resource responds to AH/gateway

10	Authorization	The AH/gateway sends a response to IH
11	Authentication	The workstation initiates mTLS to the gateway/resource for data transfer

2.2 Access Workflow

After the onboarding workflow has been completed, the IHs initiate the access workflow whenever a connection to protected workloads or resources is needed. The following steps outline the sequence of activities and interactions in the access workflow.

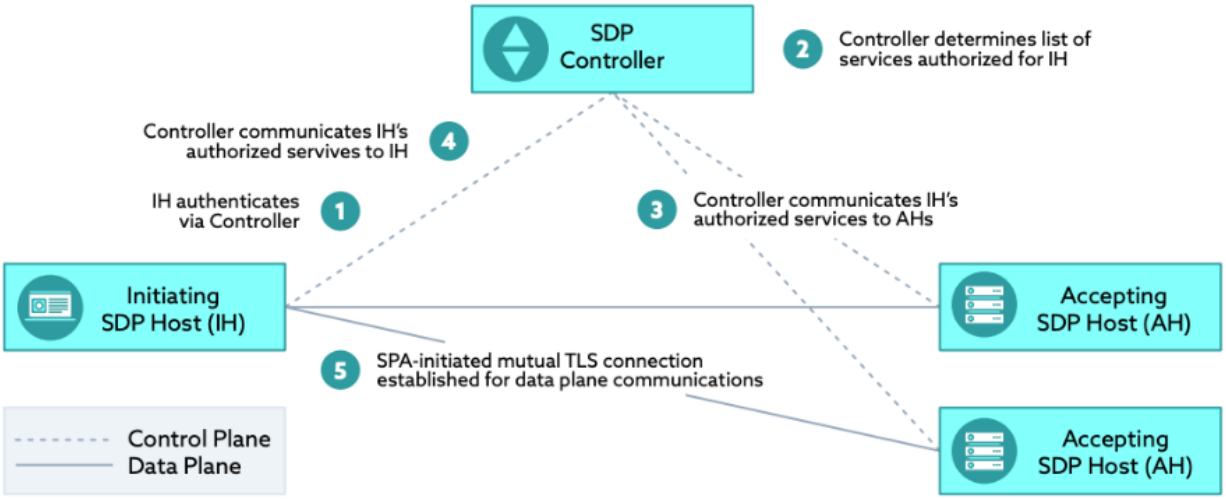


Figure 10: *Access Workflow and Interactions*

1. When an onboarded IH returns online (e.g., device reboot, user-initiated connection), it connects and authenticates to the SDP through the controller.
2. After authenticating the IH—and in some cases, querying an IdP—the controller determines a list of resources/services on the AH to which the IH is authorized access.
3. The controller instructs the AH to accept communication from the IH, providing any information that defines connectivity between users, devices, and services required for two-way encrypted communications.
4. The controller gives the IH a list of authorized AHs, as well as any additional information required for two-way encrypted communications.
5. The IH uses SPA to initiate a connection with authorized AHs, who in turn verify the SPA packets' information for enforcement. The IH creates a mutual TLS connection to those AHs.

2.2.1 Access Requests Using SPA

A core SDP principle is that all SDP components, not just protected resources/workloads, are made completely inaccessible to unauthorized entities. This is made possible by cryptographically authenticating and authorizing every entity before allowing them to connect with SDP components,

by way SPA—specifically, SDP's version of SPA, based on an RFC 4226 HMAC-based one-time password (HOTP).

The result is improved SDP security and resilience—since unauthorized entities are unable to establish a network connection with an SDP component, they cannot attempt to exploit a vulnerability, brute force a login attempt, or utilize stolen user credentials. In contrast, traditional remote access solutions like VPN must expose its components and infrastructure to all internet users, benign and malevolent.

2.2.2 Authentication

To gain access to a SDP-protected resource, the IH makes a request to the controller; in turn, the controller authenticates the user/client and determines which resources can be accessed, per policy. This determination can be based on a myriad of parameters, as the controller may obtain additional information from adjacent services (e.g., geolocation services, host validation services) to further authenticate the IH. The controller can also provide contextual information to other network components relating to the user's failing authentication or accessing of sensitive resources and services.

Authentication is typically based on user type and identity. For example, a large enterprise may have employees authenticated using an IdP, and contractors authenticated by credentials stored in a database or IAM service. The controller is the centralized point for managing and tracking the authentication process and may leverage IAM/IdP for more granular privileges. IAM/IdP provides a mechanism for storing and validating entity attributes and statuses throughout the identity lifecycle. Beyond the initial user authentication, IAM/IdP services can inform step-up authentication measures, such as prompting for an OTP to access sensitive systems, and other dynamic/adaptive security controls. Once the controller has verified that all the authentication and security posture requirements are met as part of the current access request, authorization is granted to the requested resources.

Mutual authentication between SDP components is another key aspect of SDP; per the ZT model, an organization's traffic must be encrypted across all its environments, whether they are internally or externally located. To this end, mutual TLS (mTLS) is emphasized in all the SDP deployment models and is supported by additional steps such as identity and device validation. mTLS ensures that parties at both ends of a network connection are who they claim by verifying that they both have the correct private key.

2.2.2.1 Defining Access Policies

Access policies should always follow the principle of Least Privilege—that is, access is provided to entities only as required, based on their role (e.g., user, admin, power user) and/or attributes (e.g., security posture, location, time of day). As part of a ZT environment, access policies must be dynamic, with access limited or restricted upon entry into an environment; if policies are static and/or managed manually dynamically, the full value of SDP will not be gained.

Upon authentication and authorization, the requested access is either granted or denied through policies. Once access is no longer needed (e.g., entity logs or times out), access to the service is revoked.

2.2.3 Authorization

Authorization determines what type of access is provided to requesting entities based on user roles (both current and historical), user/device attributes, or other fine-grained information related to actual data element/flow that the user is authorized for. Authorization is commonly based on group membership; however, multiple data points may be integrated within a ZTA and SDP environment to determine authorization levels. For example, to authorize a requesting entity to access a protected resource or service, the controller may leverage an internal user-to-service mapping and policy model, an IdP (e.g., Lightweight Directory Access Protocol [LDAP]) or another third-party authorization solution. In effect, the access policies maintained by the controller can be informed by other adjacent services (e.g., enterprise service directories and identity stores).

The controller enables the type of dynamic ZT policies that NIST identifies as a critical ZT tenet. Specifically, the NIST 800-207 Special Publication on ZTA prescribes a series of measures and guidelines for fulfilling the core components of ZT principles. As an example, NIST 800-207 includes the following information flows as part of the NIST Trust Algorithm for granting and denying access:

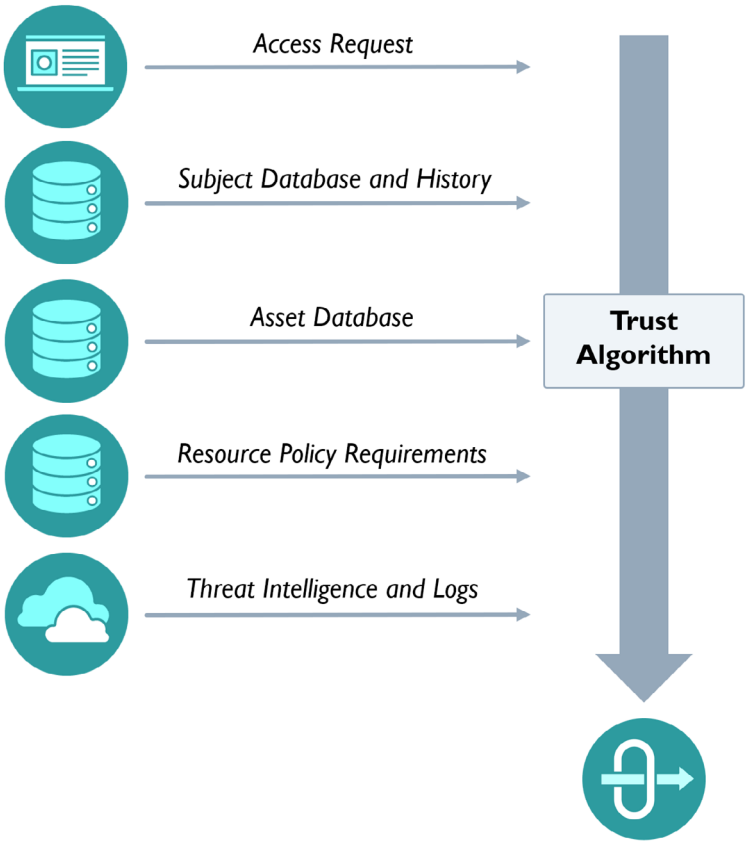


Figure 11: *NIST Trust Algorithm*[4]

[4] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf

Similarly, SDP uses multiple data sources to inform authorization decisions, including but not limited to IdP group membership, device posture, and user/device geolocation.

3 SDP Communication Flows

The following unit covers the main communication flows between SDP components, namely the IH to controller, AH to controller, and IH to AH flows, as well as the events and transactions that occur at each step of the flow.

3.1 IH to Controller Flow

The IH-controller communication flow operates on the network routing and packet delivery level, with implementation details dependent on the type of transport (e.g., TCP-based guaranteed delivery, UDP-based fire and forget).

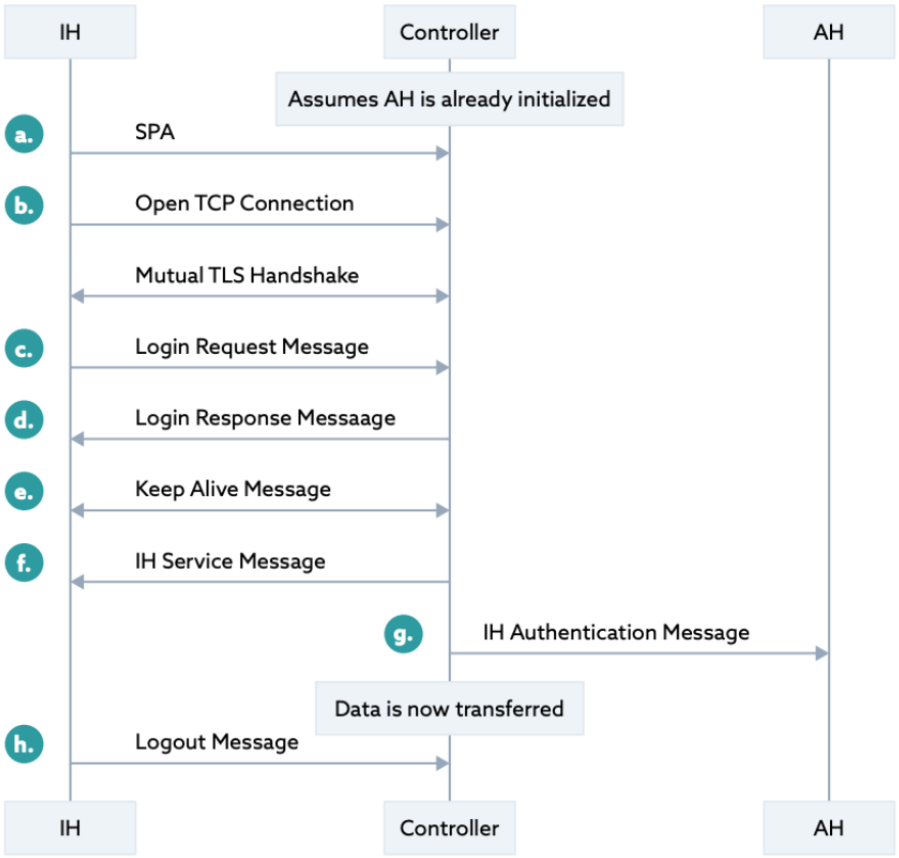


Figure 12: *Sample IH-Controller Flow*

a. The IH sends the SPA packet to the controller to request a connection

> The controller is completely hidden from the internet and allows communication only when an SPA is received by the controller from a user device or IH.

b. Open connection and establish mutually authenticated communications

> Following its sending of the SPA packet, the IH attempts to open a TCP connection to the controller. If the controller determines that the SPA packet was valid, it will permit this TCP connection to be established. This will be followed by the required mutual authentication for the establishment of an mTLS connection. In the case of UDP-based DTLS, this is a logical connection, since UDP is a connectionless protocol.

c. Login (SDP join) request message

> The login request message is sent by the IH to the controller to indicate that it is available and would like to be part of the SDP. Note that the IH login request may include credentials with which the IH identifies and authenticates itself to the controller.

d. Login response message

> The login response message is sent by the controller to indicate whether the login request was successful and, if successful, to provide the IH session ID. Note that it is possible that the controller will reject the IH's login request. This could be due to, for example, invalid credentials, or the controller could be enforcing system license or scale limits and reject this IH's attempt for those reasons.

e. Keep-alive message

> The keep-alive message is sent by either the IH or the controller to indicate that it is still active.

f. IH services message

> The services message is sent by the controller to provide the IH with the list of available services and the IP addresses or hostnames of the AHs protecting them. This message must contain sufficient information for the IH to be able to connect to the service. Note that the hostname/IP address listed is that of the AH which is directly reachable by the IH. The actual service may be running on a different host/IP than the AH. The service ID is used later by the IH to identify a target service when communicating with the AH.

g. IH authenticated message

> The role of the AH is to ensure that an authentication request is validated prior to allowing access to the list of protected resources. The controller sends the IH authenticated message to the AH to indicate to the AH that a new IH has been validated and that the AH should allow access to this IH for the specified services. Note that although this message is sent from the controller to the AH, it is initiated in response to the IH authenticating to the controller.

h. Logout request message

> The logout request message is sent by the IH to the controller to indicate that the IH wishes to terminate its SDP session. There is no response message sent by the controller, and the TLS and TCP connections must then be terminated by either the IH or the controller. Note that the IH remains onboarded, and can establish a new session again in the future.

3.2 AH to Controller Flow

The following sequence describes the AH to controller flow

a. AH sends an SPA packet to controller (SPA protects the controller from unauthorized access)
b. IH attempts a TCP connection and a subsequent mTLS connection is established
c. Login request message sent from IH to join SDP; the request may include credentials for authentication
d. Login response message is sent back by the controller
e. Logout sent by AH to the controller to indicate it's no longer available for receiving messages from the controller; TLS and TCP terminated at this point
f. Keep-alive message sent by AH or the controller
g. Service message sent by the controller to indicate the services AH is protecting

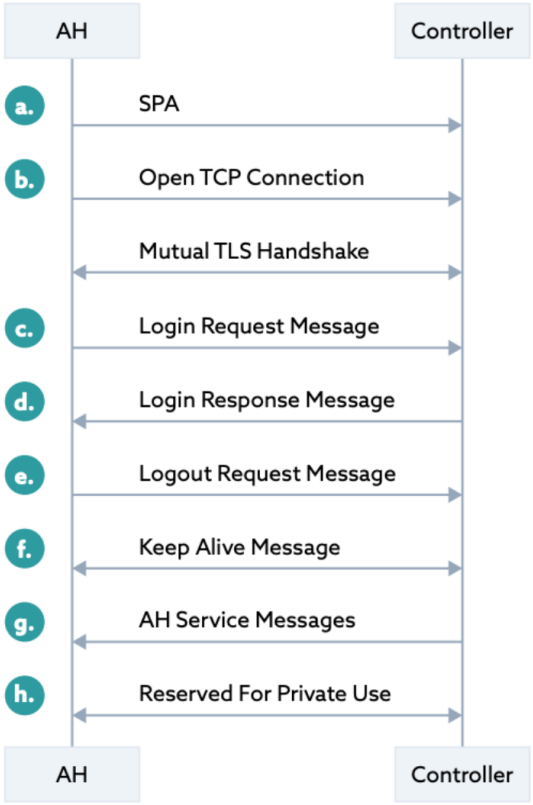


Figure 13: *Sample AH-Controller flow SDP Specification v2.0*

3.3 IH to AH Flow

The IH to AH connectivity enables access to the target applications. It is important to note that the connection is established dynamically through the AH to the service/application only after the IH has connected with a valid SPA packet, established mutually authenticated communications, and the AH verifies that the IH is authorized to access the requested service. Until then the service is kept hidden by the AH.

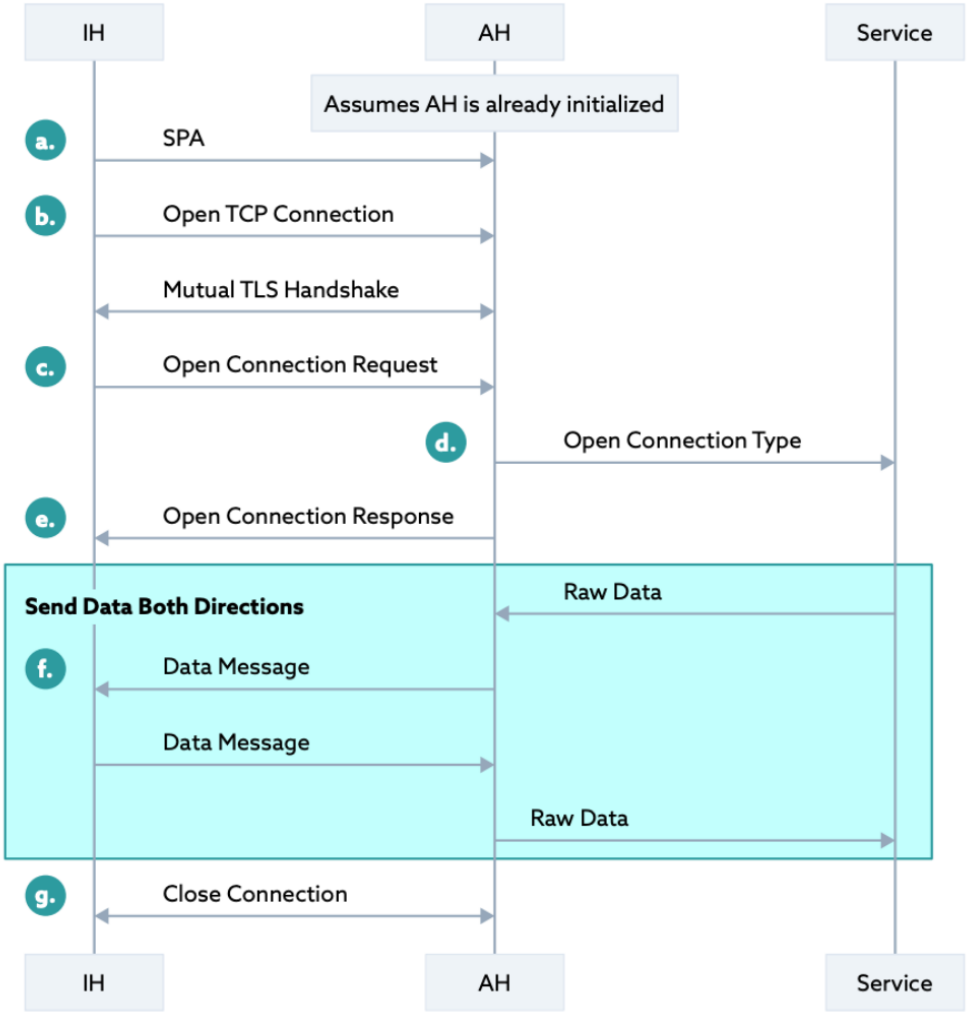


Figure 14: *Sample IH-AH flow*

a. Initiate connection request with SPA

> The SPA packet is sent by the IH to the AH to request a connection. Note that AH is completely hidden from the internet and SPA enables the visibility of AH to the IH.

b. Open connection and establish mutually authenticated communications

> After sending the SPA packet, the IH attempts to open a TCP connection to the AH. If the AH determines that the SPA packet is valid, it will permit this TCP connection to be established. This will be followed by the required mutual authentication for the establishment of an mTLS connection. In the case of UDP-based DTLS, this is a logical connection since UDP is a connectionless protocol.

c. Open connection request message

> The IH sends the open request message to the AH to indicate that it would like to open a connection to a particular service. The Service ID is a unique value assigned by the controller for

> each remote service. The IH is aware of Service IDs because they have been previously sent by the controller to the IH as part of the IH services message, depicted above in the IH-controller protocol. The IH and the AH use the Session ID to differentiate among different TCP connections for a specific remote service.

d. Open appropriate connection type

> This step, which is specific to the implementation and SDP deployment model, is used by the AH to establish a connection to the service on behalf of the IH. This may not be necessary for certain protocols, such as short-lived HTTPS connections or connections that require IH-provided application-level authentication as part of the initial exchange (e.g., SSH). SDP-aware services may need this connection to be established to communicate the IH's or user's context from the AH, outside of the application protocol. Depending on the SDP deployment model, this connection may be a network connection or a local host connection (e.g., inter-process communication).

e. Open connection response message

> The open-response message is sent by the AH to the IH to indicate whether the open request was successful. The open connection request and open connection response messages may be asynchronous for the IH, so the service ID and session ID codes are useful for the IH to associate this return value with its corresponding request.

f. Data message

> The data message is sent by either the IH or the AH. It is used to push data on an open connection. There is no response. This message is implementation-dependent. Some SDP implementations may package application data in messages such as this, while others may use alternative approaches.

g. Connection closed message

> The connection closed message is sent by either the IH or the AH. It is used to either indicate that a connection has been closed by the AH or that the IH is requesting a connection be closed. There is no response.

4 SDP Logging

Continuous logging and monitoring are crucial for ensuring that the SDP architecture's security controls are functioning as expected and intended. To effectively monitor the security of an SDP environment, each SDP architecture component creates logs detailing network activity, errors, security events, and login attempts. In turn, these logs may be forwarded to a security information and event management (SIEM) solution, security orchestration, automation, and response (SOAR) system, and/or a user entity behavior analytics (UEBA) platform for automated alerting and further analysis. By centralizing the storage and interpretation of logs for near-real-time analysis, SIEMs, SOARs, and UEBA streamline log analysis efforts and responses to security alerts generated by

applications and network components, thereby enabling security personnel to take defensive actions quickly in the event of a cyber attack.

Because SDP prescribes a highly structured approach to controlling and securing access to resources, implementations typically provide SIEMs and SOARs with even richer information than traditional network and application monitoring tools. Properly configured SDP components provide critical and detailed real-time logs correlating the who, what, and where of each connection. This level of integrated log data is not readily available in traditional network architectures, primarily due to the session encryption of most traditional network connections. Since SDP networks have visibility into mTLS traffic, they can provide additional critical logging details to third-party tools and security staff. Furthermore, integrating SIEMs and SOARs into an SDP can help manage risk in near real-time. The SIEM/SOAR/UEBA can alert the SDP controller of suspicious behavior and prompt it to drop all connections to and from a specific user or device pending further analysis.

4.1 Operations Logs

Per the ZT security model, all activity in the environment must be logged. Logging helps to ensure optimal infrastructure health, continuous resource and service availability, and an overall strong environmental security posture. The logs that are commonly generated and analyzed in an SDP architecture are as follows:

- **Operational logs**: typically contain the details of a given component's overall operations
- **Security/Connection logs**: used for troubleshooting connection issues or investigating suspicious network anomalies

Security logs in particular are critical for ensuring the strong security posture of an SDP, as they enable the detection of large-scale data breaches and cyber attacks. Both operational and security/connection logs should be forwarded to a SIEM for more extensive analyses, deeper correlations, incident detection, and proper prioritization of security alerts.

4.2 Examples of SDP Logging

The following example depicts a complete network outage involving an SDP controller failure, illustrating what log entries are created, and where they are written.

1. The IH attempts to reconnect to the controller n times and creates the relevant log entries (e.g., ops:conn:reconnect log messages)
2. After n tries, the client determines that the controller connection is down—it then attempts to find a new controller and creates the relevant log entries (e.g., ops:conn:down log message, with a severity of the error)
3. The IH connects to the newly found controller and creates the relevant log entries (e.g., ops:conn:up log message)
4. If no more controllers are available, the relevant log entries are created (e.g., ops:conn:down log message, with a severity of critical). Similarly, if a client goes down without warning (e.g., a laptop is accidentally powered off), the controller and AH would detect a failing connection and create the relevant log entries (e.g., ops:conn:down log message, with a severity of the error).

The following example depicts a user/device connecting to an SDP-protected resource (i.e., an IH connecting to AH):

1. The IH connects to the controller and the relevant log entries are created (e.g., sec:auth log message — SPA authentication to controller ops:conn:up log message)
2. The IH mutually authenticates to the controller and the relevant log entries are created (e.g., sec:connection log message)
3. The IH connects to the AH and the relevant log entries are created (e.g., sec:auth log message)

5 SDP & NIST ZTA Deployment Models

SDP's primary components and their respective workflows and communication flows were covered in-depth in the previous units. The following unit describes the configurations of these components and flows that make up the primary SDP deployment models. As an early innovator and adopter of ZTA, the NIST ZTA model is discussed first as a precursor and influencer of resulting SDP models and approaches.

The various SDP deployment models are defined by the structures of their client, server, and gateway interactions. In each model's description, a summary is provided along with benefits, most common implementation environments, and respective alignment with the NIST Zero Trust model[5]. Since use cases may vary, an exact alignment or mapping between the models may not be possible.

5.1 NIST ZTA Approach Models & Alignments

NIST SP 800-207 specifically focuses on best practices for implementing ZT, with the relevant ZTA components, associated functions, and deployment models detailed in the document. The following section includes alignments and mappings between NIST's ZTA model and CSA's SDP deployment models to better illustrate the ZT concept's cross-applicability.

Drawing from NIST SP 800-207, the following high-level diagram serves to illustrate the NIST ZTA approach to securing and accessing protected resources.

Figure 15: *NIST SP 800-207 Zero Trust Architecture - Zero Trust Access Figure*[6]

On the far left, the subject accesses a gateway (i.e., the PDP/PEP) that in turn extends the trust zone by providing access to the target resource

[5] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf
[6] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf

The following diagram depicts a mapping of SDP components to the corresponding NIST SP 800-207 architecture components.

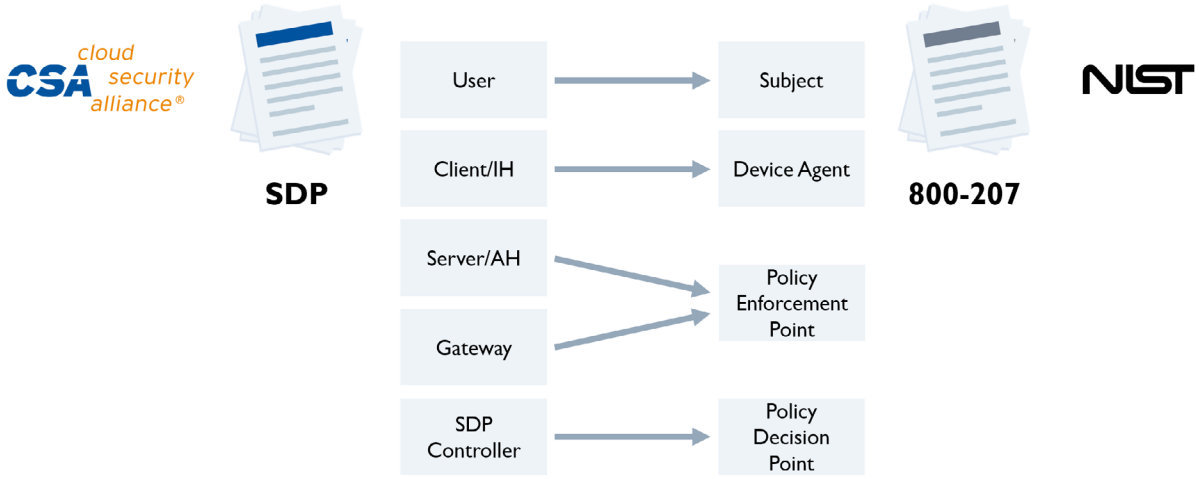


Figure 16: *SDP Component Mapping to NIST SP 800-207*

Based on these mappings, components in NIST's ZTA model clearly correspond to analogous SDP components, further illustrating how SDP can be used to meet ZT objectives and achieve interoperability between the ZTA frameworks.

The following subsections outline the four primary NIST ZTA Approach Models:

- Device Agent/Gateway-Based Deployment
- Enclave-Based Deployment
- Resource Portal-Based Deployment
- Device Application Sandboxing

5.1.1 Device Agent/Gateway-Based Model

In the device agent/gateway-based deployment model, an agent sits on, or directly in front of, the enterprise system and serves as a proxy for managing all connections. The agent communicates with the gateway and policy administrator to determine access to a given resource. The Policy Administrator and Policy Engine evaluate access requests to determine what privileges are granted, if any.

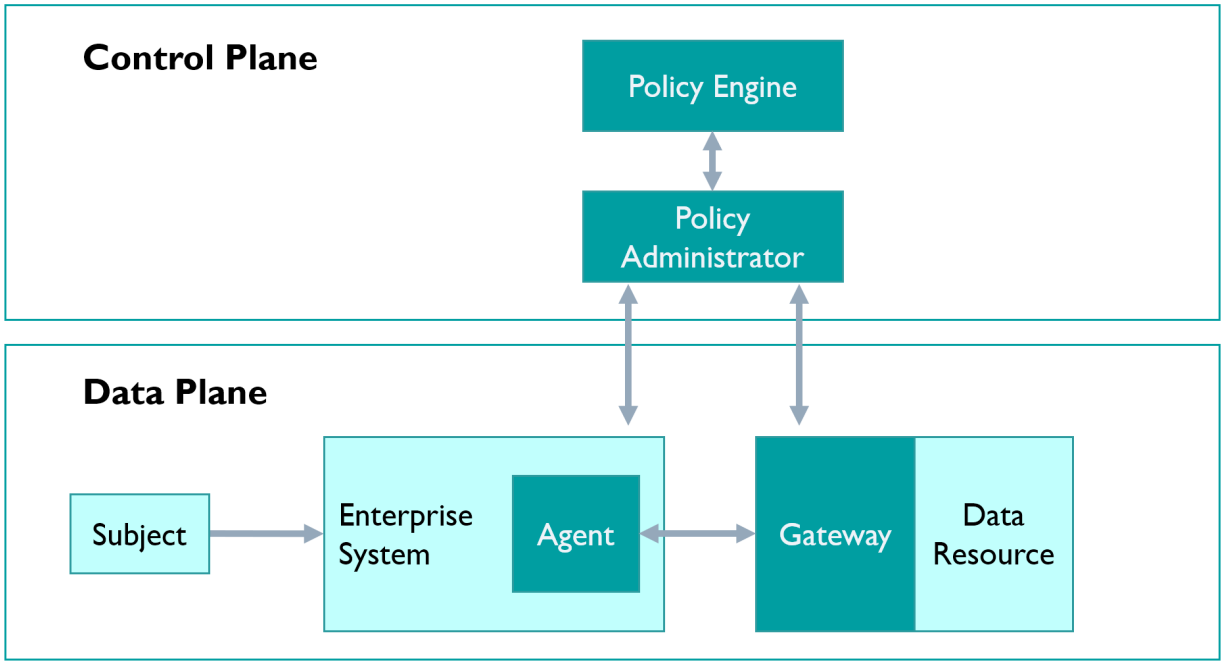


Figure 17: *NIST SP 800-207 Device Agent/Gateway Model*

This deployment model is ideal for:

- Modern corporate environments
- Managed client devices
- Typical compute/server environments or other similar use cases

5.1.2 Enclave-Based Model

The enclave-based deployment model is similar to the device agent/gateway model, with the exception that multiple resources are protected behind a single gateway. This gateway functions as a boundary for the resources it controls access to.

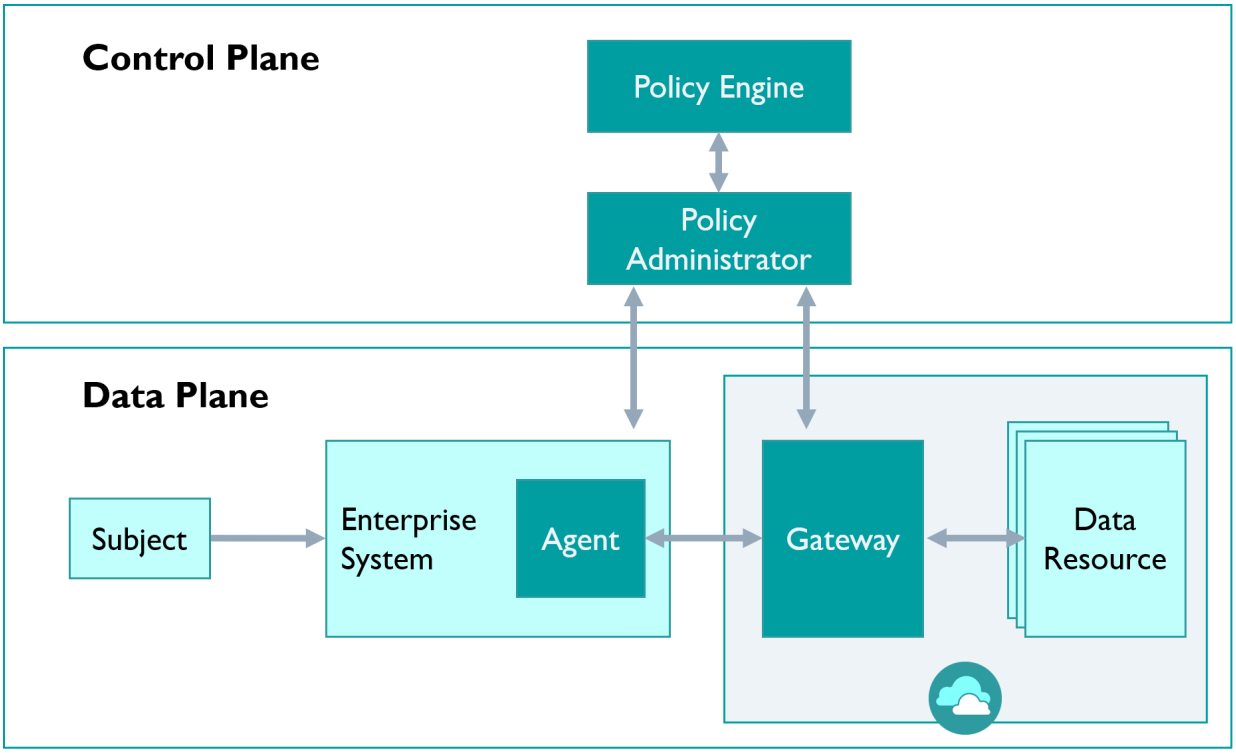


Figure 18: *NIST SP 800-207 Enclave Gateway Model*

This deployment model is ideal for:

- Microservice environments supporting a single business service
- Services which do not support the local gateway option

5.1.3 Resource Portal-Based Model

The resource portal-based deployment model removes the agent function from a system and centralizes the functions to a gateway portal. Subsequently, all gateway and agent functions seen in previous models are now centralized and contained on the portal, enabling any system with the appropriate permissions to access protected data resources.

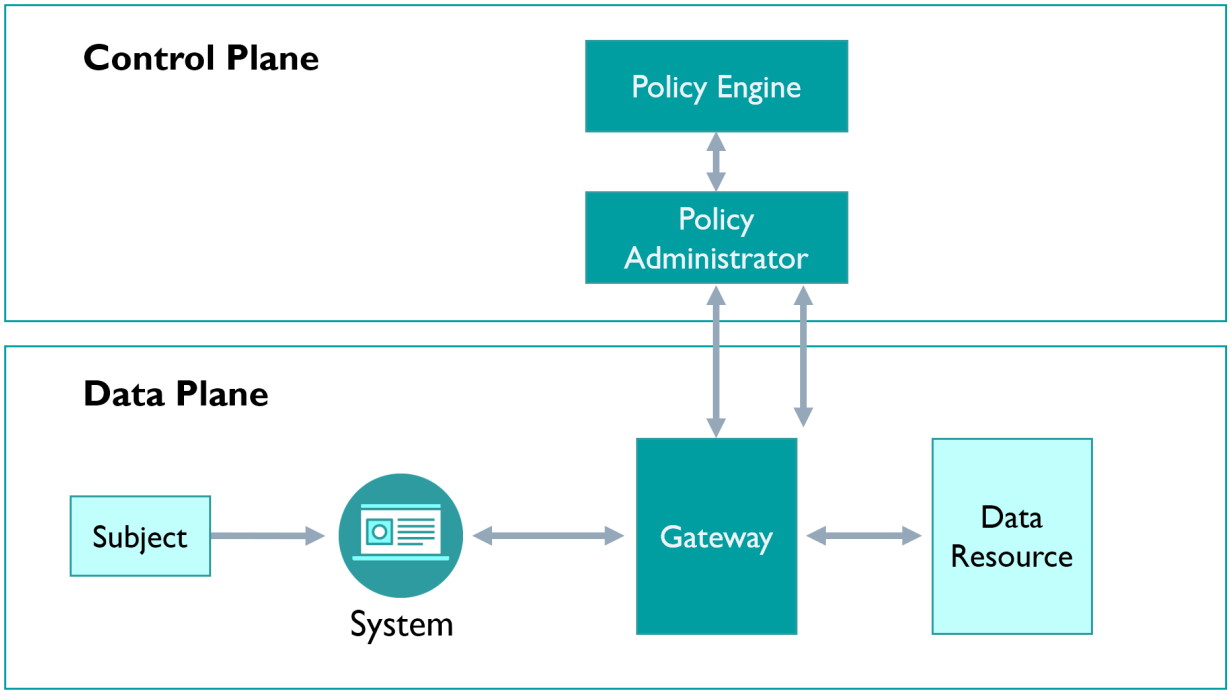


Figure 19: *NIST SP 800-207 Resource Portal Model*

This deployment model is ideal for:

- Mature BYOD environments
- Legacy environments
- Environments with black box servers (e.g., ICS/OT, SCADA, systems) and/or resources with limited management capabilities

5.1.4 Device Application Sandboxing Model

The device application sandboxing model uses trusted applications residing on a resource to communicate with a PEP. A gateway is no longer used to manage connections to resources—instead, all access and authorization decisions are carried out with a PEP and vetted application. Additionally, applications are only run in a sandbox environment and generally do not have access to local asset resources.

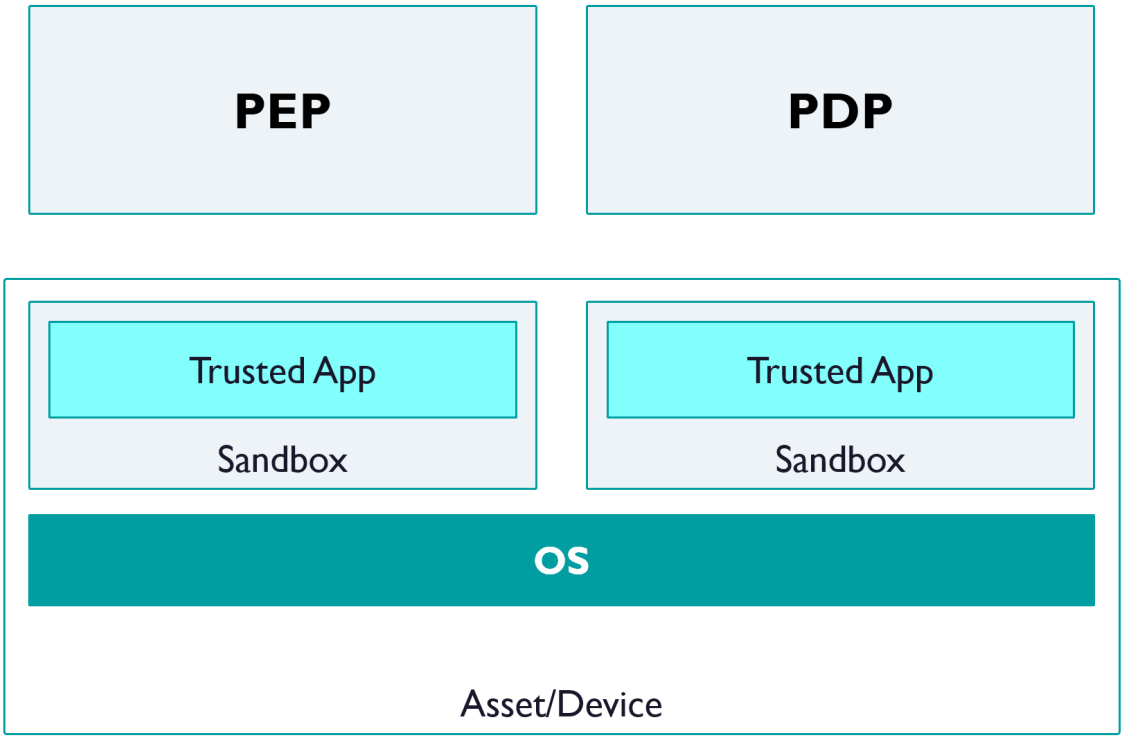


Figure 20: *NIST 800-207 Application Sandboxing Model*

This deployment model is ideal for:

- Mature BYOD environments
- Microservice and containerized environments
- Environments where assets cannot be fully managed to meet ZTA requirements

5.2 SDP Deployment Models

The following six SDP deployment models use different network topologies to secure access to protected resources, with each model providing specific benefits depending on the environment. For example, the gateway-to-gateway model is ideal for environments that cannot support a local agent/software on the IH and AH.

SDP deployment model selection should include a comprehensive evaluation of the environment's capabilities and risk tolerance—efforts that should involve both business stakeholders as well as IT

and security groups. Depending on the protected resources in question, different SDP deployment models may pose increased or decreased risk to business functions.

5.2.1 Client-to-Gateway Model

The client-to-gateway model creates a gateway between the protected servers/services and the IH by positioning servers behind the AH, with multiple services/servers protected behind the gateway. This model mitigates the risk of man-in-the-middle (MITM) attacks, scanning, and lateral movement, since requested resources and/or services are completely obfuscated from the external environment. By forcing all connections through a gateway/AH, the client-to-gateway model allows direct exposure of the gateway/AH to the internet. Subsequently, the model is ideal for securing both cloud-based applications and legacy, on-premise applications, as well as environments (e.g., proprietary server/services, PaaS/SaaS) that cannot be addressed by discrete resource models (e.g., client-to-server, server-to-server).

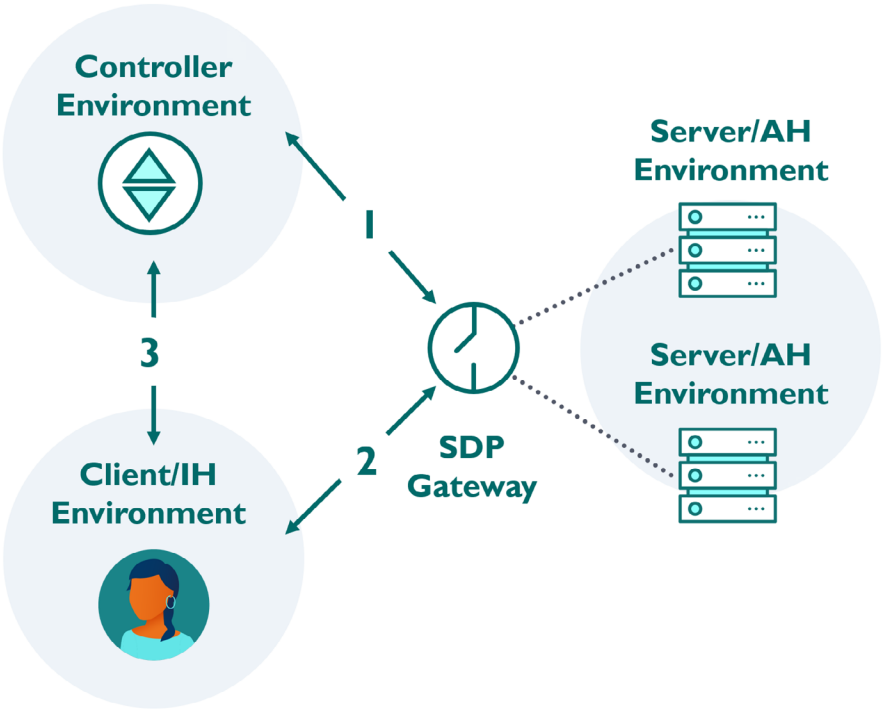


Figure 21: *SDP Client-to-Gateway Model*

This model closely aligns with the NIST enclave-based deployment model, where resources are protected behind a gateway. The main difference between the NIST enclave-based deployment model and the SDP client-to-gateway model is that multiple resources are protected by the same gateway instance, which generally does not reside on a resource. From a component level, the controller generally acts as the PDP for managing policies and supporting information. The client is the subject with an installed agent and communicates through the gateway, the PEP, which validates the connection request and allows access to appropriate resources.

5.2.2 Client-to-Server Model

SDP's client-to-server model is similar to NIST ZTA's device agent/gateway model; however, in the case of the client-to-server model, the server that runs the AH software is protected, not the gateway. The client-to-server model is typically implemented to protect cloud-based applications.

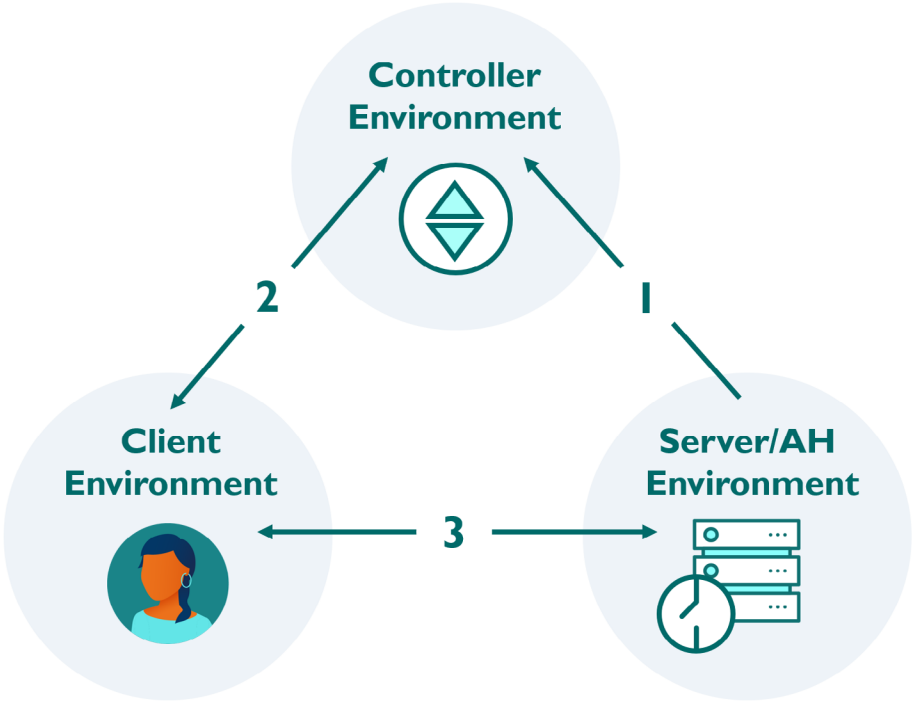


Figure 22: *SDP Client-to-Server Model*

The following considerations can help determine when to select the client-to-server model over the device agent/gateway model:

1. Load balancing requirements
2. Server elasticity
3. Number of servers requiring protection

As mentioned previously, the SDP client-to-server model aligns closely with the NIST ZTA device agent/gateway model; in this configuration, a data resource is protected by a local gateway residing on the resource. The client-to-server model is also similar to the enclave model—though the client-to-server model uses separate gateways for each protected resource, allowing for more explicit controls. At the component level, the client-to-server model is almost identical to the client-to-gateway model, with the exception that the client-to-server model's gateway/PEP protects a smaller number of resources and is usually a locally-installed agent.

5.2.3 Server-to-Server Model

The SDP server-to-server model is similar to the NIST ZTA device agent/gateway model; however, the SDP server-to-server model's IH can also function as an AH for receiving connections/requests, with the SDP gateway acting as an AH on each server. The server-to-server model is most appropriate for protecting cloud, OT, IoT, and virtual machine (VM) environments.

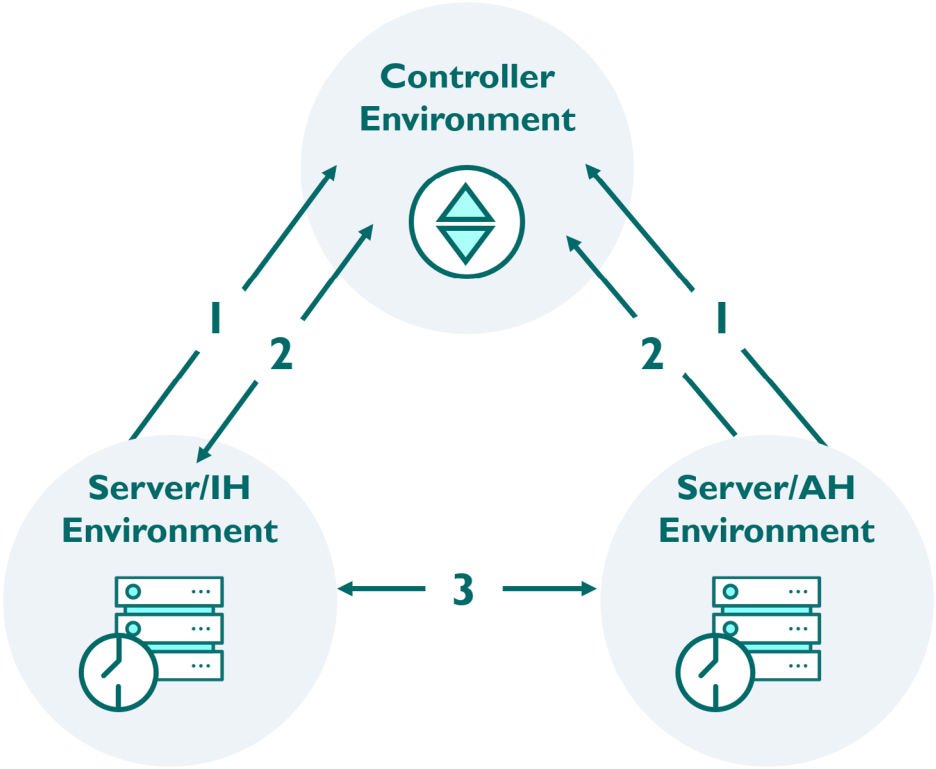


Figure 23: *Server-to-Server Model*

In contrast, the NIST ZTA deployment models do not explicitly cover bi-directional use cases addressed by the SDP server-to-server model. Per NIST, each server aligns with the gateway (i.e., PEP) and protected resources. Depending on the protected resource in question, the NIST ZTA device agent/gateway model or enclave-based deployment model may be implemented.
The SDP server-to-server model's subject and resource will shift, depending on the information flow; aside from this difference, the SDP server-to-server model shares the same components as the NIST ZTA device agent/gateway model and enclave-based deployment models.

5.2.4 Client-to-Server-to-Client Model

The SDP client-to-server-to-client model functions by designating the server as the intermediary between clients, thereby obfuscating the IP addresses of the connecting clients. This deployment model is dependent on the peer-to-peer (P2P) connection between clients and is commonly used for protecting applications such as chat, video conferencing and IP telephony.

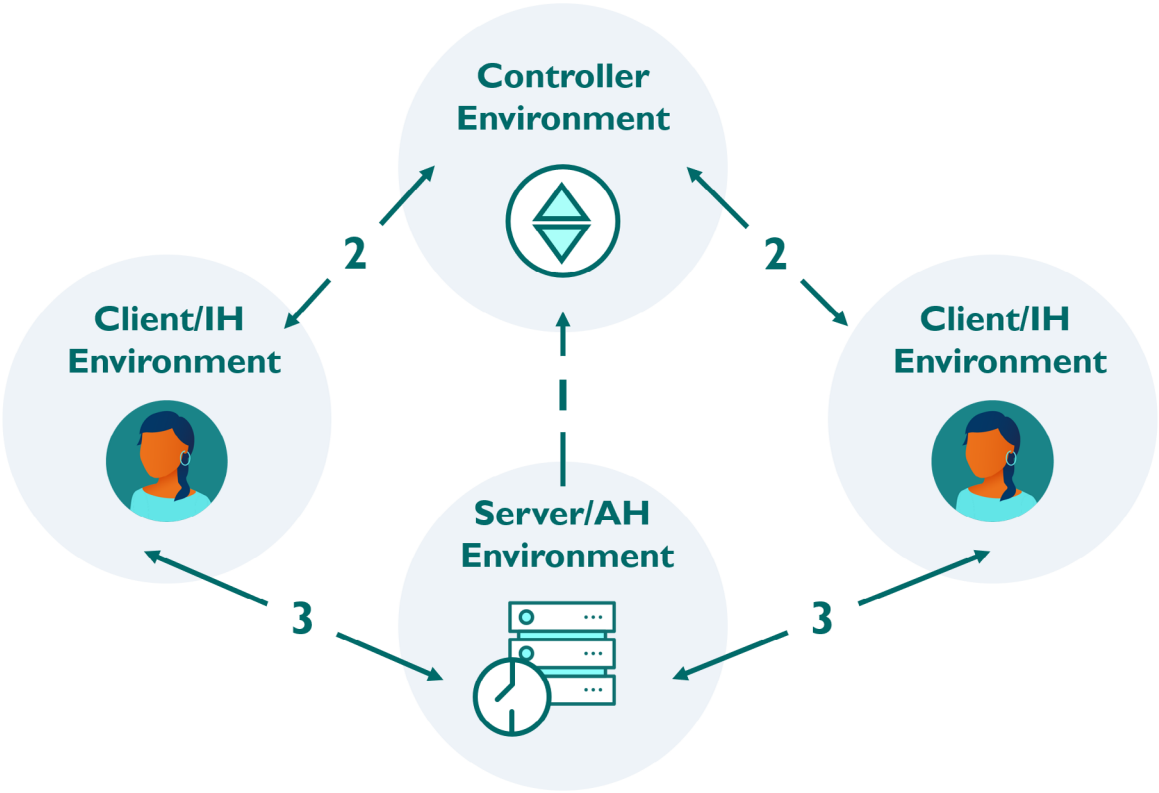


Figure 24: *SDP Client-to-Server-to-Client Model*

In the context of NIST ZTA, the client-to-server-to-client model is a combination of two device agent/gateway models, each with the same data flows and components.

5.2.5 Client-to-Gateway-to-Client Model

The SDP client-to-gateway-to-client model is similar to the SDP client-server-client model and is also dependent on P2P, with each client acting as the IH, AH, or both when connecting to each other. The SDP client-to-gateway-to-client model is typically used for securing client-to-client communications.

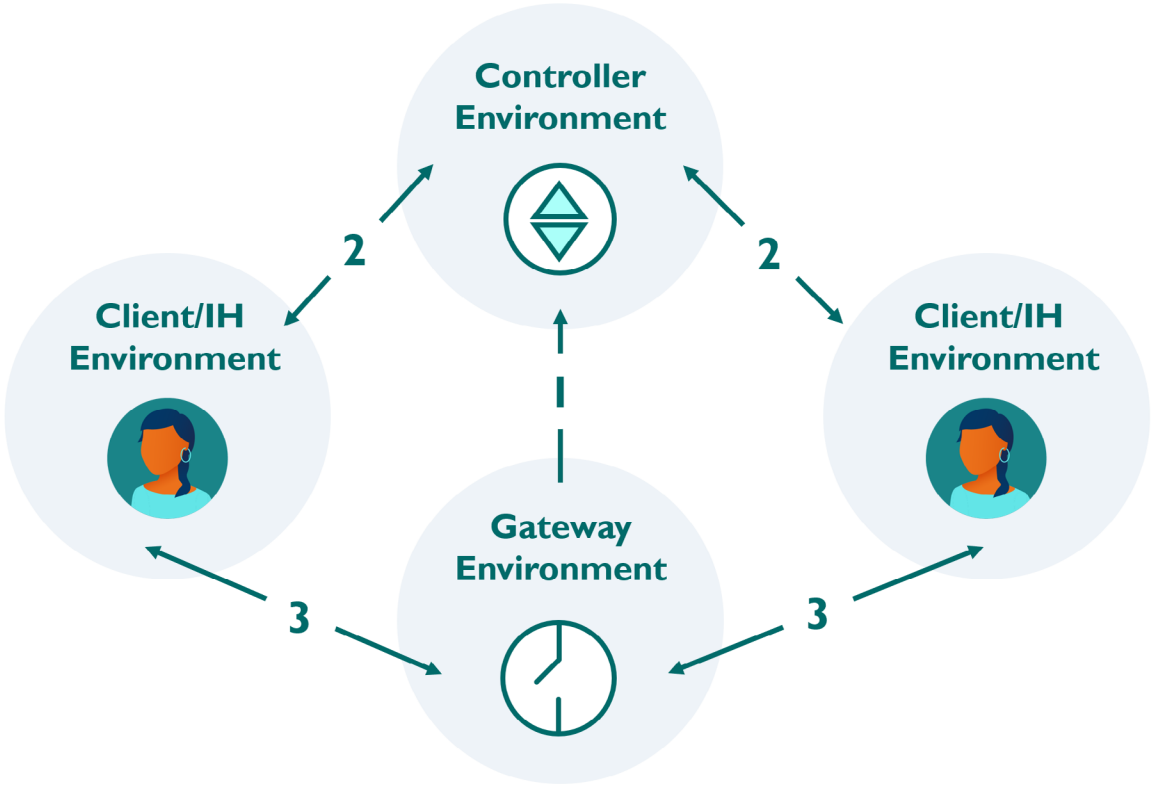


Figure 25: *SDP Client-to-Gateway-Client Model*

Depending on the gateway implementation, the client-to-gateway-to-client model generally aligns with either the NIST ZTA enclave-based deployment model or resource portal-based model. If authentication is handled through a local agent, the client-to-gateway-to-client implementation aligns more closely to the NIST ZTA enclave-based deployment model, with components swapping places based on the data flow (e.g., left-hand client [subject] accesses right-hand client [resource] through the gateway [PEP]).

5.2.6 Gateway-to-Gateway Model

The SDP gateway-to-gateway model functions by positioning one or more servers behind the AH, with the AH functioning as the gateway. One or more clients can be positioned behind the IH, effectively using it as the gateway.

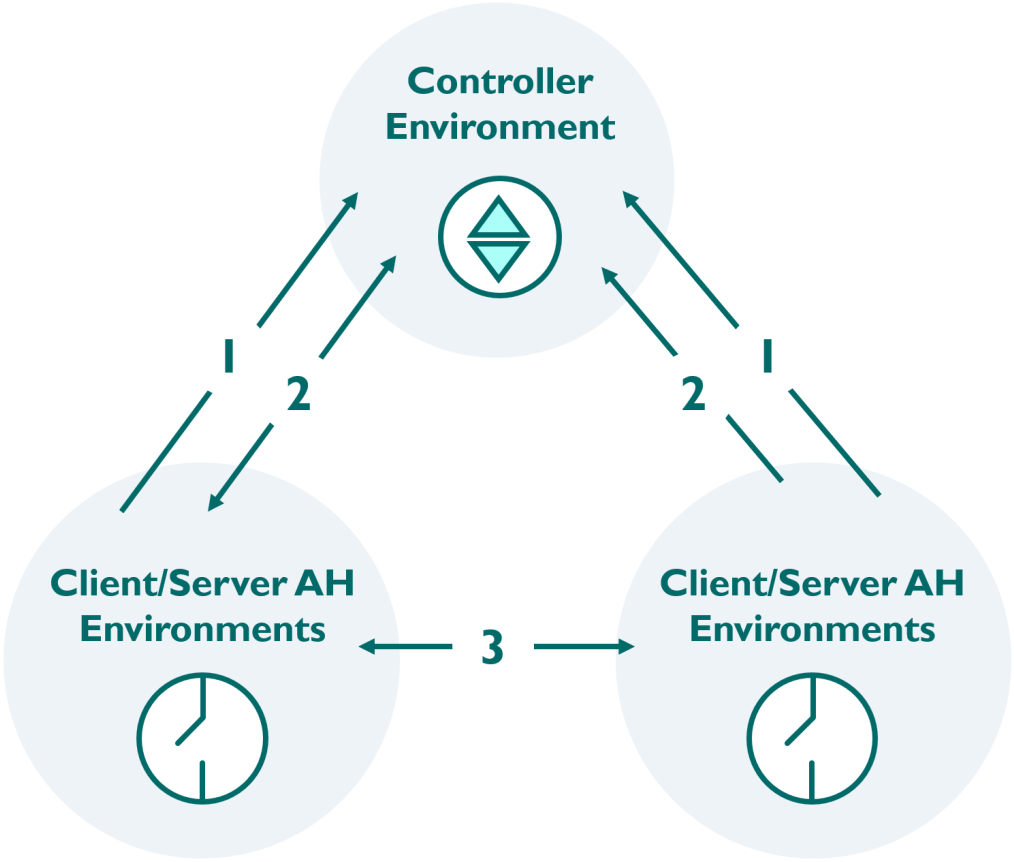


Figure 26: *SDP Gateway-to-Gateway Model*

Depending on how the gateway (i.e., PEP) is implemented, the SDP gateway-to-gateway model may align with either the NIST ZTA device agent/gateway model or enclave-based deployment model. In general, the components in the SDP gateway-to-gateway model remain the same, but may move locations from a resource to a dedicated, separate PEP.

Conclusion

In this course, learners were provided with an overview of SDP's primary components, followed by the main SDP onboarding and access workflows. The communication flows between SDP components were discussed, as well as the SDP logging functions and types for monitoring the security of the SDP environment. Lastly, learners were exposed to SDP deployment models and their respective alignments with NIST ZT deployment models.

Glossary

For additional terms, please refer to our Cloud Security Glossary, a comprehensive glossary that combines all the glossaries created by CSA Working Groups and research contributors into one place.

Term	Definition	Source
802.1x	An IEEE standard for local and metropolitan area networks–Port-Based Network Access Control. IEEE 802 LANs are deployed in networks that convey or provide access to critical data, that support mission critical applications, or that charge for service. Port-based network access control regulates access to the network, guarding against transmission and reception by unidentified or unauthorized parties, and consequent network disruption, theft of service, or data loss.	https://1.ieee802.org/security/802-1x/
Accepting Host (AH)	The SDP policy enforcement points (PEPs) that control access to any resource (or service) to which an identity might need to connect, and to which the responsible enterprise needs to hide and control access. AHs can be located on-premises, in a private cloud, public cloud, etc.	https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-zero-trust-specification-v2/
Access	To make contact with one or more discrete functions of an online, digital service.	https://csrc.nist.gov/glossary/term/access
Active Directory (AD)	A Microsoft directory service for the management of identities in Windows domain networks.	https://csrc.nist.gov/glossary/term/active_directory
Air-Gapped Networks	An interface between two systems at which (a) they are not connected physically and (b) any logical connection is not automated (i.e., data is transferred through the interface only manually, under human control).	https://csrc.nist.gov/glossary/term/air_gap
Application Programming Interface (API)	A system access point or library function that has a well-defined syntax and is accessible from application programs or user code to provide well-defined functionality.	https://csrc.nist.gov/glossary/term/application_programming_interface

Attribute-Based Access Control (ABAC)	An access control approach in which access is mediated based on attributes associated with subjects (requesters) and the objects to be accessed. Each object and subject has a set of associated attributes, such as location, time of creation, access rights, etc. Access to an object is authorized or denied depending upon whether the required (e.g., policy-defined) correlation can be made between the attributes of that object and of the requesting subject.	https://csrc.nist.gov/glossary/term/abac
Authentication	Verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in an information system.	https://csrc.nist.gov/glossary/term/authentication
Authorization	The right or a permission that is granted to a system entity to access a system resource.	https://csrc.nist.gov/glossary/term/authorization
Brute Force Attacks	An attempt to discover a password by systematically trying every possible combination of letters, numbers, and symbols until you discover the one correct combination that works.	https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks
Cloud Access Security Broker (CASB)	On-premises, or cloud-based security policy enforcement points, placed between cloud service consumers and cloud service providers to combine and interject enterprise security policies as the cloud-based resources are accessed. CASBs consolidate multiple types of security policy enforcement.	https://www.gartner.com/en/information-technology/glossary/cloud-access-security-brokers-casbs
Control Plane	Used by various infrastructure components (both enterprise-owned and from service providers) to maintain and configure assets; judge, grant, or deny access to resources; and perform any necessary operations to set up communication paths between resources.	https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf

Controller (SDP Controller)	Determines which SDP hosts can communicate with each other. The controller may relay information to external authentication services such as attestation, geo-location, and/or identity servers.	https://downloads.cloudsecurityalliance.org/initiatives/sdp/Software_Defined_Perimeter.pdf
Data Plane	Used for communication between software components. This communication channel may not be possible before the path has been established via the control plane.	https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf
Distributed Denial-of-Service (DDoS)	Involves multiple computing devices in disparate locations sending repeated requests to a server with the intent to overload it and ultimately render it inaccessible.	https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1800-15.pdf
Firewall	An inter-network connection device that restricts data communication traffic between two connected networks. A firewall may be either an application installed on a general-purpose computer or a dedicated platform (appliance), which forwards or rejects/drops packets on a network. Typically firewalls are used to define zone borders. Firewalls generally have rules restricting which ports are open.	https://csrc.nist.gov/glossary/term/firewall
Gateway (SDP Gateway)	Provides authorized users and devices with access to protected processes and services. The gateway can also enact monitoring, logging, and reporting on these connections.	https://cloudsecurityalliance.org/artifacts/sdp-architecture-guide-v2/
Hypertext Transport Protocol Secure (HTTPS)	A secure network communication method, technically not a protocol in itself, HTTPS is the result of layering the Hypertext Transfer Protocol (HTTP) on top of the SSL/TLS protocol, thus adding the security capabilities of SSL/TLS to standard HTTP communications.	https://iapp.org/resources/article/hypertext-transfer-protocol-secure/
Identity (ID)	The set of attribute values (i.e., characteristics) by which an entity is recognizable and that, within the scope of an identity manager's responsibility, is sufficient to distinguish that entity from any other entity.	https://csrc.nist.gov/glossary/term/identity

Identity and Access Management (IAM)	The set of technology, policies, and processes that are used to manage access to resources.	https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-203.pdf
Identity Provider (IdP)	A trusted entity that issues or registers subscriber authenticators and issues electronic credentials to subscribers. A cloud service provider may be an independent third party or issue credentials for its own use.	https://csrc.nist.gov/glossary/term/identity_provider
Initiating Host (IH)	The host that initiates communication to the controller and to the AHs.	https://downloads.cloudsecurityalliance.org/initiatives/sdp/SDP_Specification_1.0.pdf
Lightweight Directory Access Protocols (LDAP)	A networking protocol for querying and modifying directory services running over TCP/IP.	https://csguide.cs.princeton.edu/email/setup/ldap
Man-in-the-middle (MITM) attacks	An attack where the adversary positions himself in between the user and the system so that he can intercept and alter data traveling between them.	https://csrc.nist.gov/glossary/term/mitm
Micro-segmentation	Is the technique of creating secure zones within a data center and cloud deployments that allow the organization to separate and secure each workload. This makes network security more granular and effective. These secure zones are created based on business services, and rules are defined to secure information workflow.	https://www.techtarget.com/searchnetworking/definition/microsegmentation
Multi-factor Authentication (MFA)	Authentication using two or more factors to achieve authentication. Factors include: (i) something you know (e.g., password/personal identification number (PIN)); (ii) something you have (e.g., cryptographic identification device, token); or (iii) something you are (e.g., biometric).	https://csrc.nist.gov/glossary/term/multi_factor_authentication
Mutual Transport Layer Security (mTLS)	An approach where each microservice can identify who it talks to, in addition to achieving confidentiality and integrity of the transmitted data. Each microservice in the deployment has to carry a public/private key pair and uses that key pair to authenticate to the recipient microservices via mTLS.	https://cheatsheetseries.owasp.org/cheatsheets/Microservices_security.html#mutual-transport-layer-security

Network Access Control (NAC)	A method of bolstering the security of a private or "on-premise" network by restricting the availability of network resources to endpoint devices that comply with a defined security policy.	https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-41r1.pdf
Network Segmentation	Splitting a network into sub-networks, for example, by creating separate areas on the network which are protected by firewalls configured to reject unnecessary traffic. Network segmentation minimizes the harm of malware and other threats by isolating it to a limited part of the network.	https://www.nist.gov/itl/smallbusinesscyber/cybersecurity-basics/glossary
Open Systems Interconnection (OSI)	Qualifies standards for the exchange of information among systems that are "open" to one another for this purpose by virtue of their mutual use of applicable standards.	https://www.ecma-international.org/wp-content/uploads/s020269e.pdf
Phishing	A technique for attempting to acquire sensitive data, such as bank account numbers, through a fraudulent solicitation in email or on a web site, in which the perpetrator masquerades as a legitimate business or reputable person.	https://csrc.nist.gov/glossary/term/phishing
Policy decision point (PDP)	Mechanism that examines requests to access resources, and compares them to the policy that applies to all requests for accessing that resource to determine whether specific access should be granted to the particular requester who issued the request under consideration.	https://csrc.nist.gov/glossary/term/policy_decision_point
Policy enforcement point (PEP)	A system entity that requests and subsequently enforces authorization decisions.	https://csrc.nist.gov/glossary/term/policy_enforcement_point
Port	Another essential asset through which security can be breached. In computer science, ports are of two types - physical ports (which is a physical docking point where other devices connect) and logical ports (which is a well-programmed docking point through which data flows over the internet). Security and its consequences lie in a logical port.	https://www.w3schools.in/cyber-security/ports-and-its-security/

Public Key Infrastructure (PKI)	The framework and services that provide for the generation, production, distribution, control, accounting, and destruction of public key certificates. Components include the personnel, policies, processes, server platforms, software, and workstations used for the purpose of administering certificates and public-private key pairs, including the ability to issue, maintain, recover, and revoke public key certificates.	https://csrc.nist.gov/glossary/term/public_key_infrastructure
Role Based Access Control (RBAC)	Access control based on user roles (i.e., a collection of access authorizations a user receives based on an explicit or implicit assumption of a given role). 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.	https://csrc.nist.gov/glossary/term/role_based_access_control
Security Assertion Markup Language (SAML)	A protocol consisting of XML-based request and response message formats for exchanging security information, expressed in the form of assertions about subjects, between online business partners.	https://csrc.nist.gov/glossary/term/security_assertion_markup_language
Security Orchestration Automation and Response (SOAR)	Refers to technologies that enable organizations to collect inputs monitored by the security operations team. SOAR tools allow an organization to define incident analysis and response procedures in a digital workflow format.	https://www.gartner.com/en/information-technology/glossary/security-orchestration-automation-response-soar
Single Packet Authorization (SPA)	Can authenticate a user to a system for simple remote administration. It is a protocol for allowing a remote user to authenticate securely on a "closed" system (limited or no open services) and make changes to or run applications on the "closed" system.	https://www.blackhat.com/presentations/bh-usa-05/bh-us-05-madhat.pdf

Software-Defined Network (SDN)	An approach to computer networking that allows network administrators to manage network services through abstractions of higher-level functionality. SDNs manage the networking infrastructure. This is done by decoupling the system that makes decisions about where traffic is sent (the control plane) from the underlying systems that forward traffic to the selected destination (the data plane).	https://ieeexplore.ieee.org/abstract/document/6819788
Software-Defined Perimeter (SDP)	A network security architecture that is implemented to provide security at Layers 1-7 of the OSI network stack. An SDP implementation hides assets and uses a single packet to establish trust via a separate control and data plane prior to allowing connections to hidden assets.	https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-and-zero-trust/
Transmission Control Protocol (TCP)	A transport protocol that is used on top of IP to ensure reliable transmission of packets. TCP includes mechanisms to solve many of the problems that arise from packet-based messaging, such as lost packets, out of order packets, duplicate packets, and corrupted packets. Since TCP is the protocol used most commonly on top of IP, the Internet protocol stack is sometimes referred to as TCP/IP.	https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:transporting-packets/a/transmission-control-protocol--tcp
Transmission Control Protocol/ Internet Protocol (TCP/IP)	A set of protocols covering (approximately) the network and transport layers of the seven-layer Open Systems Interconnection (OSI) network model.	https://www.gartner.com/en/information-technology/glossary/tcpip-transmission-control-protocolinternet-protocol
Transport Layer Security (TLS)	A cryptographic protocol, successor to SSL, that provides security for communications over a computer or IP network.	https://csrc.nist.gov/glossary/term/transport_layer_security
Virtual Private Network (VPN)	A virtual network built on top of existing physical networks that can provide a secure communications mechanism for data and IP information transmitted between networks or between different nodes on the same network.	https://csrc.nist.gov/glossary/term/virtual_private_network