Key Features & Technologies 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:

Abhishek R. Singh
Heinrich Smit
Jacob Kline
Matthew Meersman, PhD
Michael J. Herndon
Michael Roza
Prasad T.
Vani Murthy

Contributing Editors:

Alex Sharpe
Alexander Stein
Ashwini Siddhi
Dr. Amol Khedgikar
James Lam
Magdy Elfaramawy
Rajesh Ingle, PhD
Remo Hardeman
Reto Kaeser
Richard Lee
Sam Reddy
Shinesa Cambric

Expert Reviewer:

Clément Betacorne
Farid F. Gurbanov
Gustavo Vallejo
Ledy M. Eng
Madhav Chablani
Naresh Kurada
Peter van Eijk
Ron Martin (Dr.), PhD
Ronald Kearns
Shruti Kulkarni

CSA Staff:

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

Table of Contents

List of Figures and Tables

Course Intro

Welcome to SDP Key Features & Technologies by the Cloud Security Alliance (CSA). This training module is part of a larger series of CSA programs focused on Zero Trust (ZT) and Zero Trust Architecture (ZTA). As a security model for achieving ZT supported by extensive CSA research, SDP provides organizations with a flexible, vendor-agnostic approach to protecting IT infrastructures from increasingly sophisticated cyber threats. In this course, learners will get an in-depth look at the key SDP features and technologies for securing today's and tomorrow's IT infrastructures—whether they are on-premises, in the cloud, or a hybrid of the two, including cases of multiple cloud service providers.

Course Structure

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

1. Traditional architectural challenges
2. The principles of least privilege and need to know
3. Hiding of infrastructure
4. Mutual authentication
5. Tunneling
6. Micro-segmentation
7. Identity and access management
8. Secure remote access
9. Software-defined networking and Software-Defined Perimeter

Course Learning Objectives

After completing this course, learners will be able to:

- Understand the limitations of traditional security architecture when integrating security controls, securing shifting perimeters, and dealing with the IP address challenge
- Explain SDP's key features, including separate data and control planes, default drop-all firewalls, and single packet authorization
- Discuss how policy-based authorization and granular access controls help enforce the principle of least privilege and need to know
- Distinguish between SDP's secure remote access and virtual private network
- Describe mutual authentication, tunneling, micro-segmentation, and identity and access management in the context of SDP
- Compare and contrast software-defined networks and SDP

1 Traditional Architectural Challenges

By providing organizations with a practical framework and set of technologies to achieve ZT, SDP enables organizations to largely mitigate the security risks inherent in traditional architectures. This section outlines several key issues and concerns with traditional architectures that are addressed by SDP, namely, the complexity of integrating security controls, the shifting perimeter challenge, and the IP address challenge.[1]

1.1 Complexity of Integrating Security Controls

Complexity is an enemy of security. Integrating security controls is difficult with traditional perimeter-based security and defense-in-depth approaches; more often, these require a combination of tools and techniques that require a considerable amount of specialized knowledge and numerous skill sets. The successful deployment of firewalls, intrusion detection/prevention systems (IDS/IPS), VPNs, and other security tools in concert with each other can be difficult, error-prone, and expensive. Maintenance and changes are continuously needed after the various systems' security alarms and responses have been coordinated. Additionally, using systems from multiple vendors, if not implemented carefully, can increase risk rather than reduce it. Lastly, maintaining and patching systems can lead to increased governance efforts, and hamper rapid response efforts.

1.2 The Shifting Perimeter Challenge

Virtualized, disparate network segments have all but replaced the traditional fixed network perimeter in which trusted internal network segments are protected by network appliances (e.g., load balancers and firewalls). The increasing number of mobile and Internet of Things (IoT) devices and the global shift to remote work make the fixed network perimeter model increasingly ineffective. Organizations responding to the needs and expectations of remote employees typically adopt bring-your-own device (BYOD) policies and expand secure remote access connectivity options. The cloud further complicates the situation as organizations adopt public and private clouds using multiple cloud service providers with multiple service delivery models (i.e., IaaS, PaaS, SaaS).

We will address the new security paradigm and building analogy in the following sections.

1.2.1 New Security Paradigm Challenges

The increasing heterogeneity of IT environments has resulted in rich, sizable targets for malicious actors wishing to inflict harm on the organization. Unfortunately, to secure networks using the traditional approaches, all network traffic must be funneled back to fixed perimeters, resulting in significant bottlenecks and inefficiencies. Additionally, traditional fixed network perimeters are less effective since their inherent network security models grant privileges based on an entity's location with respect to the fixed network perimeter (i.e., either inside or outside the perimeter). Once fixed perimeter defenses are breached, intruders can easily access IT assets and devices on the internal network.

[1] Cloud Security Alliance, "SDP Architecture Guide v2," 7th, May 2019, https://cloudsecurityalliance.org/artifacts/sdp-architecture-guide-v2/

Traditional defense-in-depth practices are also ineffective at hindering determined attackers, as they typically use a fragmented approach to risk mitigation that leaves gaps in network security. For example, an overly complex environment employing numerous security solutions increases human errors and management overhead, leading to misconfigurations and administrative noise (e.g., an abundance of false positive security alerts). The resulting security glitches make it difficult for security teams to identify/prioritize the most serious threats, thereby giving malicious actors more attack vectors to work with.

1.2.2 The Building Analogy

Traditional network security implementations are analogous to doors and walls designed to protect physical property. Access to interior rooms is open, as long as potential visitors have keys to the external door lock. If malicious actors manage to pick the exterior doors they have free run of the building.

Today, organizations typically rely on their digital front door locks with continuous monitoring to ensure that cyber criminals fail to break in. In such a model, authenticated users, once connected to the internal network, are authorized to access a wide variety of systems on the network. A malicious actor can penetrate the network by bypassing a proverbial single door lock, allowing him to pivot to additional systems and escalate privileges. Lateral movements within a fixed perimeter like these is a longstanding challenge.

1.3 The IP Address Challenge

The Transmission Control Protocol/Internet Protocol (TCP/IP) was designed in an era when the prevailing networking model was connect first, authenticate second. This model implies a certain degree of trust between clients and servers, which is not always deserved. Critically, this presents malicious actors with the opportunity to exploit vulnerabilities they would otherwise not, had the connection never been established to begin with.

The following sections provide a more in-depth explanation regarding the shortcomings of the TCP/IP's connect first, authenticate second security model.

1.3.1 Access Before Authentication

Traditional security mechanisms may provide strong external defenses against malicious actors, but they still allow entities to connect to the network prior to authentication, offering little protection against insider threats. For example, firewalls are typically configured to allow or deny specific IP addresses via allowlists and denylists; however, devices behind the firewalls are assumed trustworthy. Malicious attackers already inside the network have access to the IT assets in the environment, even without express permission to access specific services or resources.

Similarly, VPNs are usually configured to only allow users with authorized VPN clients and appropriate keys onto the network. However, a cyber attacker that manages to clone a VPN client and steal its keys can access privileged network resources, move laterally, and perform malicious activities such as credential theft and denial of service (DoS) on other systems. Once VPN users are authenticated, they are assumed to be trustworthy and have unfettered access and visibility on the network.

1.3.2 Broad Network Connectivity & Exposure

Public/private cloud infrastructures emulate on-premises data centers, and inherit both their strengths and weaknesses. Like on-premises data centers, public/private clouds implement perimeter network security using a layered approach (e.g., streaming logs to monitoring tools and leveraging hybrid security controls). However, these security features do not address the connect first, authenticate second problem. Users and devices, once connected, have broad access to IT assets and services across the internal environment, even without explicit privileges.

1.3.3 Lack of Granular Access Mechanisms

Traditional security mechanisms like VPNs and firewalls do not use explicit, fine-grained network access controls. Network layer firewalls are static and rely only on network information—it is therefore not unusual for users from varying departments with different roles to require access to the same service with the same IP address. This is problematic, as different users and devices present varying risks that require more granular and/or controlled access levels. Traditional firewalls typically do not adjust their rules based on the context of the request (i.e., the level of trust granted to a given device/network).

A common scenario involves users requesting access to the organization's environment over an unsafe connection (e.g., a free wifi hotspot at an airport). A traditional network firewall cannot detect if the local machine's endpoint protection and/or antivirus software has been turned off by malware or by accident. Even if VPN is used, the connection still poses considerable risk. For example, Internet Protocol Security (IPSec) VPNs are reliant on tokens and credentials vulnerable to interception. Similarly, Secure Sockets Layer (SSL) VPNs have been known to have exploitable vulnerabilities.

1.3.4 Shortcomings of Traditional Firewall Architectures

Traditional firewalls are designed to deny unauthorized access attempts and permit authorized access based upon a set of pre-defined organizational security rules and policies. The prevailing methodology for configuring traditional firewalls is centered around static rules and policies defining the opening of specific ports and protocols for specific applications. For example, because the Domain Name System (DNS) typically relies on TCP port 53, an associated static firewall rule would be required for authorizing communications to external DNS resources through that port.

This rule-based approach is problematic for numerous reasons. In traditional IT environments, servers and network devices can be easily scanned using a wide range of tools (e.g., nmap, Zmap, Nessus). This enables both administrators and malicious actors to interrogate networks for specific devices and running services. Closed or filtered ports on a server or network device may still share details regarding its hosted applications and operating system.

Additionally, traditional firewalls, next generation firewalls (NGFWs), and unified threat management (UTM) systems inspect packets as they pass through the device, effectively serving as a choke point to control network traffic flow. Firewall devices use rules based on the source/destination of the packets for determining whether to allow or deny the traffic. The capacity of the firewall is therefore the limiting factor—if network traffic exceeds the firewall's capabilities, the firewall can fail open or

closed. If it fails open, all additional traffic goes uninspected. If it fails closed, no additional traffic is allowed until the traffic rate drops within the firewall's capacity, effectively causing a DoS. More complex rules are possible, but resulting solutions quickly become complex to maintain, process, and analyze.

Another fundamental limitation of traditional firewalls is that they cannot inspect or protect network traffic that doesn't pass through the organization's internal network (e.g., VPN users directly accessing an internet resource). To overcome this shortcoming, administrators may force all of its user traffic from outside the firewall to flow back through the organization's internal network, and then out through the firewall to internet or cloud destinations, thereby ensuring that all relevant traffic is inspected. Unfortunately, this so-called hairpin configuration causes significant network latency when large groups of employees work remotely and are accessing the organization's IT resources located in cloud environments. Additionally, this creates new attack vectors for malicious actors to exploit when trying to compromise the organization's IT infrastructure.

1.4 How SDP Addresses Traditional Architecture Shortcomings

To mitigate the challenges related to traditional architectures, an SDP architecture uses specialized security controls to enforce ZT. For example, micro-segmentation, drop-all firewalls, and single packet authorization (SPA) address issues with shifting perimeters and TCP/IP's connect first, authenticate second model, as well as broad network connectivity and lack of access level granularity. Additionally, SDP can augment and/or replace VPN as a more secure method for remote access.

The following units of this module discuss SDP's key features, capabilities, and technologies, and delve into greater detail regarding how they address the challenges and limitations of traditional architecture.

2 The Principles of Least Privilege & Need to Know

The principles of least privilege and need to know are foundational concepts of ZT and SDP that, when enforced with user access controls and authentication/authorization procedures, ensure that only authenticated/authorized entities are granted access to information or systems based on their specific job function. SDP mandates that protected resources are completely hidden from requesting entities and made visible only after they have been authenticated and authorized. Chiefly, SDP accomplishes this by limiting access to required applications, applying granular access controls at the element level, and employing policy-driven authorization and access to expand or restrict an entity's privileges.

2.1 Access & Visibility Granted on an As-Needed Basis

Per the principles of least privilege and need to know, an entity is granted network visibility and access to an application on an as-needed basis. In ZT, these restrictions are typically managed with logical access controls. For example, users assigned to human resources (HR) roles may be granted access to a cloud-based people management platform; however, the devices used to access the platform may be restricted to only United States HR employees. Limiting access to the bare minimum required to perform the duties at hand is crucial for protecting assets and limiting the impact radius of cyber failures; as duties change, access controls should also dynamically adjust accordingly (e.g., disable access).

2.2 Granular Controls at the Application Level

Policy assignment for users/user groups is typically performed within the SDP administration toolset, which is in turn synchronized with other authorization platforms and services (e.g., identity providers [IdPs] such as Lightweight Directory Access Protocol [LDAP] and OpenID). This allows for granular access controls at the application level, in accordance with the principles of least privilege and need to know. Depending on the access granted by the policy, additional authentication steps may be necessary to validate the user's identity (e.g., password re-confirmation, multi-factor authentication, validation of a device's media access control [MAC] address). Organizations may also incorporate attribute-based access controls (ABAC) for more granular controls at the application level; unlike role-based access controls (RBAC), ABAC evaluates attributes versus roles when making access decisions.

2.3 Granular Controls at the User Level

In addition to application level controls, user level controls can also be leveraged to enforce the principles of least privilege and need to know. SDP agents installed on user devices can provide seamless enforcement of controls used in securing the accessed data. For example, user level controls may restrict/block the copying and pasting of sensitive data and/or use geolocation to determine a user's access level and privileges.

2.4 Policy-Driven Authorization & Access

SDP relies on policy-driven authorization and access to implement access control based on the principles of least privilege and need to know, thereby allowing entities to only access what they require to complete the task on hand. Policy-driven authorization and access uses attributes to define policies that specify access permissions to resources. Policies allow or limit a user, user group, or role's privileges to carry out actions.

The following key attributes are typically defined in policies and are used during authorization and privilege assignment processes:

- User type (e.g., human users, accounts used by systems and applications)
- Job categories the user belongs to in the organization (e.g., HR, operations, legal)
- Action the user is allowed to perform (e.g., view, edit, share, or delete data)
- Target resource or object (e.g., specific endpoints, entire databases, shared folders)
- User credentials required to access the resource at the action level (e.g., initial credentials are sufficient, re-authentication required, additional authentication steps required)

2.4.1 Dynamic Policies

Dynamic policies allow and deny access to protected resources based on dynamically generated authorization decisions. The decisions are made using pre-configured policies versus static rules, enabling organizations to leverage cybersecurity and operational enhancements that address some of the critical shortcomings of traditional firewalls. Policies managed at the SDP controller can be

used to specify the details and conditions for dynamically opening/closing firewall ports for specific IP addresses, as well as define user/user group access privileges to applications. This includes access restrictions based on geography, network, or device type, to name a few.

By allowing organizations to create and implement highly specific firewall rules, dynamic policies offer key benefits such as a reduced attack surface, the integration of multiple security data sources for better-informed access decisions, time-bound connections for automatically revoking/restricting access based on connection duration, and an overall reduction in administrative overhead and management complexity.

2.4.1.1 Reduced Attack Surface

Policy-driven authorization and access reduces the organization's attack surface by implementing dynamic, policy-based decisions and instituting the principle of least privilege and need to know. Policies can be resource-specific and configured to apply highly granular access controls to protected assets, resulting in stricter criteria-based access control policies for protecting a resource. The result is more tightly managed, better controlled access to resources based on an entity's unique criteria; in contrast, traditional firewall rules allow broad level access that makes for a significant attack surface.

2.4.1.2 Integrated Security Data Sources

Dynamic policies can be designed to use aggregated inputs from multiple sources (e.g., alerts from IDS/IPS, threat intelligence, network device health, netflow event monitoring) to dynamically inform access control decisions. By correlating various security events in a threat-informed approach, organizations are more adequately equipped to defend themselves from evolving cyber threats. Additionally, integrated security data sources allow organizations to dynamically pivot and enhance the policies driving the access control decisions for individual resources. This automatic enhancement of access control policies based on integrated security intelligence from the target environment is a substantial benefit of dynamic policies.

2.4.1.3 Time-Bound Connections

Dynamic policies also incorporate time-bound connectivity for controlling port access, allowing for IT resources to be shared with a given entity for precisely the amount of time required. For the duration of the connection, the IT infrastructure is completely hidden from the internet and authenticated users' port access is limited to the session duration of the application. Once the timeout threshold is reached, the connection is terminated and the protected infrastructure is again hidden from the initial requester.

2.4.1.4 Reduced Administrative Burden

As dynamic policies are by definition more flexible than static rules, the operational burden related to managing static rulesets is significantly reduced. This benefit is primarily a result of policy reuse as applied to individual protected resources, as the latter can be grouped with others requiring similar access control protections, thereby reducing policy maintenance overhead. Though their policies may be initially adapted from static firewall rules, dynamic firewalls require significantly less maintenance over a longer period of time.

3 Hiding of Infrastructure

Any service that responds to incoming network packets— even in rejecting or selectively accepting requests—can be abused during malicious reconnaissance efforts (i.e., obtaining information about the protected assets and/or network infrastructure behind the service). Additionally, all technologies, whether hardware, software/services, or a combination thereof, have vulnerabilities or flaws that may eventually result in a compromised security posture. This is even true of security devices like firewalls and secure network protocols such as SSL or Secure Shell (SSH). For these reasons, all IT infrastructure components should be hidden by default from unauthorized and unauthenticated entities.

Security controls should be placed where packets enter the network which is configured to ignore traffic from all IP addresses except those belonging to authenticated and authorized entities. An SDP architecture accomplishes this by hiding an organization's assets behind an SDP gateway, the component providing isolation and access controls for the services logically behind it.

Because SDP architectures completely hide an organization's infrastructure from the internet, non-authenticated and non-authorized users/devices are unable to access protected infrastructures or even know of their existence. This hiding of infrastructure is a crucial defense, as malicious actors cannot attack what they cannot discover or see; SDP provides this capability while at the same time providing legitimate end users with seamless and frictionless access to the organization's assets and resources.

In the past, infrastructure hiding was partially achieved through port knocking, or use of a daemon or service on a port knocking server with packet filtering to monitor firewall logs. Specifically, the daemon or service would look for a predefined sequence of client attempts to access closed ports; once a port knock sequence is recognized, the client IP address is authorized for the requested access.

Unfortunately, port knocking is both computationally and resource intensive, as it requires both packet encryption as well as compensatory mechanisms for out-of-sequence knocks (e.g., the result of unpredictable network routes). Port knocking is also limited to sending two bytes of data per packet, since it uses the port headers which, in TCP and User Datagram Protocol (UDP), are only 16-bits wide; in fact, port knocking effectively creates a DoS by adding noise to knock packets. Lastly, more sophisticated attackers can record and replay port knock sequences in man-in-the-middle (MITM) attacks. These port knocking drawbacks and limitations make SDP a more elegant and effective solution for hiding IT infrastructure.

The three main technologies that enable the hiding of infrastructure in an SDP are:

- Separate data and control planes
- Default drop-all firewall
- SPA

The mechanics behind each of these technologies are discussed more in-depth in the following sections.

3.1 Separate Data & Control Planes

In alignment with the ZT model, an SDP architecture uses two separate layers of traffic to ensure that entities cannot see or access the organization's network until they are authenticated and authorized. The control plane manages how data is forwarded and is responsible for establishing controls like granular segmentation, while the data plane handles the actual forwarding.

The following diagram illustrates how data and control plane traffic is separated in an SDP implementation.

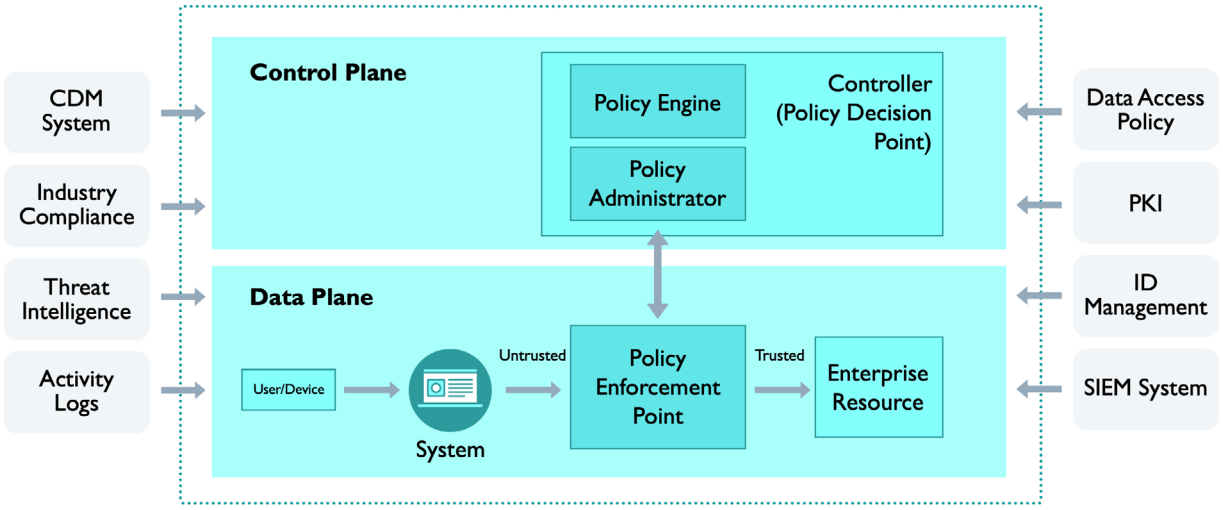

Figure 1: *Separate Data and Control Planes*[2]

The connecting user's device establishes two different planes of traffic in order to secure access to the application. As a first step, the user and their device's identity are authorized and authenticated at the control plane; during this step, multi-factor authentication (MFA) can be incorporated to provide an extra layer of security. The connecting user's device will then connect to the controller, followed by the control plane validating the user and device and facilitating application connectivity via the data plane. No users or devices access the data plane without permission from the controller; once authenticated and authorized, requesting entities establish connections to the IT assets in question via the data plane. The controller shares the connection details to construct the SPA for the user's device, which in turn uses it to establish connectivity to the accepting host (AH).

3.2 Default Drop-All Firewall

A firewall's default policy determines how to proceed when network traffic does not match any existing rules. By default, the firewall can either accept or deny any unmatched traffic; a default accept (i.e., deny list) policy means that any unmatched traffic is implicitly allowed into the network, whereas a default deny (i.e., allow list) or ''drop-all'' policy means that any traffic not explicitly allowed will be dropped. A default drop-all posture is generally more secure, as it requires less management and guarantees that any existing traffic on the network is explicitly permitted.

[2] Figure adapted from NIST, "SP 800-207 Zero Trust Architecture," August 2020, https://csrc.nist.gov/publications/detail/sp/800-207/final

For this reason, an SDP architecture incorporates the default drop-all firewall as an integral component for automatically dropping any unsolicited network packets received. This constitutes the most reliable firewall policy for ensuring that all the organization's assets are hidden from the internet. By ensuring that all traffic aimed at the AH is discarded, the default drop-all firewall effectively hides the AH's presence—even port sniffing tools (e.g., nmap, Netcat) cannot detect or identify the AH's open ports. The drop-all firewall creates the illusion that the protected assets do not exist on the network while providing visibility and connectivity for authenticated devices both dynamically and selectively.

3.3 Single Packet Authorization (SPA)

SPA is a crucial mechanism used by SDP to ensure that both the organization's IT resources as well as SDP infrastructure components are completely inaccessible to unauthenticated and unauthorized entities. As discussed previously, the default drop-all firewall ensures that all the organization's IT assets are hidden from view unless access is explicitly granted. SPA is the primary mechanism that enables this hiding of infrastructure.

SPA packet creators (i.e., initiating host [IH]) and recipients (i.e., AH) are required to establish a shared root of trust, as shared secrets known only to the parties involved are required when constructing valid SPA packets. The establishment of this root of trust—that is, how the shared secret is securely communicated to SDP components—depends on the implementation; typically, this information is included during the onboarding process for IHs and AHs. Entities in possession of the shared cryptographic secret can generate a valid SPA packet and connect to the organization's network and respective IT assets.

While the drop-all firewall drops all incoming packets, a second service or daemon (e.g., fwknopd) uses a passive method (e.g., libpcap, tcpdump) to pick up the dropped packets for SPA. Once a valid SPA packet with data is identified, the firewall is instructed to allow the requestor appropriate access to the resource for the required duration, per the defined policies. On the other hand, dropped packets gathered at SDP gateways can be forwarded to third-party tools like security information and event management systems (SIEM), IDS/IPS, and security orchestration, automation and response (SOAR) platforms for further analysis. Because the first packet an AH receives from any other host must be a SPA packet, if any other packet type is received, it should be viewed as an attack. Therefore, the SPA enables the SDP to determine an attack based on a single malicious packet.

3.3.1 UDP Versus TCP-Based SPA

Depending on the chosen SDP implementation, SPA packets may be initiated using either UDP or TCP. Architects are well-advised to consider the strengths and weaknesses of each protocol carefully, as the benefits will vary depending on their respective IT infrastructures' composition. For example, TCP is a session-based protocol that consists of an initial handshake (i.e., synchronize [SYN], synchronize-acknowledgement [SYN-ACK], acknowledge [ACK]) preceding the transmission of data, followed by a similar handshake (i.e., FIN, FIN-ACK, ACK) to close the connection. Because of its session-based nature, TCP connections are far less likely to experience packet loss in data flows and is recommended for applications where packet delivery must be guaranteed. Unfortunately, TCP also has vulnerabilities (e.g., TCP session hijacking) that make it a popular target for exploitation by cyber attackers.

Additionally, SDP components using TCP-based SPA expose open ports to all remote and potentially malicious entities; subsequently, infrastructure hiding is not completely achieved. Assets are also partially exposed to potential DoS attacks, as they will likely permit the establishment of a TCP connection from any remote IP address, and then perform SPA validation prior to creating a TCP/TLS connection. Using SPA with TCP will permit the server to detect an attack based on an invalid SPA packet, but only after the TCP connection is established, thereby consuming server resources.

In contrast, UDP packets do not maintain sessions, and packets are sent to a destination without any confirmation of receipt. Packet loss is therefore more common; subsequently, UDP should only be used when a data flow can accept some packet loss. However, because no session creation/dismantling is required, UDP is faster and causes less overhead—benefits ideal for low-bandwidth environments.

When deciding between UDP and TCP, the architect must take into account the aforementioned security impact but also the importance of the service(s) being protected by SPA. If a SPA failure presents risk, it may be best to utilize TCP. If the risk is lower from a service availability perspective then UDP should better protect the environment.

3.3.2 SPA Message Format

Each SPA packet is composed of a series of mandatory and optional fields that together contain all the necessary information for authorizing the connection. The SDP client installed on the IH accepts user input, populates the SPA packet with data from this input, encodes and encrypts the packet, and applies a hash-based message authentication code (HMAC)-based one-time password (HOTP) algorithm before sending out the packet request. This enables the service/daemon to passively monitor the default drop-all firewall's discarded traffic, validate any incoming SPA packets, and apply the appropriate access controls.

Validating incoming SPA packets is computationally lightweight, yet highly effective in making SDP systems more resilient against DoS attacks. Because unauthorized and unauthenticated entities cannot establish a network connection with SDP components, cyber attackers cannot brute force a login attempt or utilize stolen user credentials to gain access. In contrast, traditional remote access solutions like VPNs expose the network to all entities on the Internet—including malicious actors. While SPA message formats may differ between SDP implementations, all SDP systems should support SPA as the mechanism for initiating connections between components.

The following table describes the format used in SPA messages.

ClientID	This is a 256-bit numeric identifier, assigned per user-device pair. This field is used to distinguish the user, device, or logical group that is sending the packet.
Nonce	This is a 16-bit random data field that prevents replay attacks by avoiding SPA packet reuse. Also referred to as packet numbers or initialization vectors, the nonce can only be used once in a cryptographic communication.

Timestamp	This field prevents servicing outdated SPA packets, by ensuring a short time period of validity (for example, 15 to 30 seconds). This also provides a mechanism to reduce the replay-detection caching required for the recipient.
Source IP Address	This is the publicly visible IP address of the IH. This is included so that the AH does not rely on the source IP address in the packet header, which is easily modified en route. The IH must be able to obtain the IP address for use by the AH as the originator of the packets.
Message Type	This field is optional—it may be used to inform the recipient what type of message to expect from the IH after the connection is established.
Message String	This field is optional and will be dependent on the Message Type field. For example, this field could be used to specify the services that an IH will be requesting if known at connection time.
HOTP	This hashed one-time-password is generated by an algorithm as described by RFC 4226, based on a shared secret. The use of an one-time password (OTP) is required in SPA packets for authenticity; other OTP algorithms can be substituted with the overarching goal of providing authenticity of the SPA packet.
HMAC	This field is calculated over all fields above. Algorithm choices are SHA256 (recommended), SHA384, SHA512, SM3, Equihash, or other efficient and robust algorithms. The HMAC is calculated using a shared seed and is generated over all prior fields of the message, then used by the AH to verify message integrity. The HMAC validation is computationally lightweight and therefore resilient against DoS attacks. Any SPA packets with invalid HMACs will be immediately discarded.

Table 1: *SPA Message Scheme*[3]

Though SPA options may include additional encryption (e.g., using the IH's private key for non-repudiation, or the AH's public key for confidentiality), asymmetric encryption is computationally expensive and should only be used by the recipient following a more lightweight validation mechanism (e.g., simple HMAC) to keep the AH resilient against DoS attacks.

3.3.3 SPA Benefits

SPA is capable of substantially mitigating the risk of existing and future attacks with minimal server resources and management overhead/complexity, making it the ideal complement to SDP. The following list describes the key benefits that SPA brings to an SDP architecture.

[3] Table adapated from Cloud Security Alliance, "Software-Defined Perimeter (SDP) Specification v2," 10th, March, 2022, https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-zero-trust-specification-v2/

- **Hides SDP system components:** Neither SDP controllers nor AHs will respond to any connection attempts (e.g., TCP SYN packets sent as connection requests) until the remote system has provided an authentic SPA packet valid for that SDP system. This prevents the disclosure of connection information to a potential attacker and works for standalone AHs as well as AHs that are logically part of a server/workload.

- **Mitigates DoS attacks on TCP/Transport Layer Security (TLS):** Internet-facing servers running Hypertext Transfer Protocol Secure (HTTPS) are highly susceptible to DoS attacks. SPA mitigates these attacks by allowing the server to quickly reject unauthorized connection attempts, prior to incurring the overhead of establishing a TCP/TLS connection.

- **Enables attack detection:** The first packet sent to an AH from any other host must be a SPA packet. If an AH receives any other packet, the event is immediately regarded as an attack. SPA enables the SDP to accurately identify attacks based on a single malicious packet.

- **Designed for flexibility:** SDP architectures allow for diversity in potential approaches and deployment models. For example, SDP has six deployment models, each addressing different problems through separate implementation methods with varying tradeoffs.

- **Allows for a self-contained security architecture:** Once seeded, IHs are equipped to securely and reliably establish encrypted communications with controllers and AHs, without relying on any external systems.

SPA also provides additional benefits as a secure, self-contained, connectionless message transmission protocol. For example, a SPA packet can be used as a way to securely transmit data from a remote endpoint. Because its packets are based on shared secrets, SPA recipients can trust that the data was issued by a valid SDP client. Since the SPA seed (i.e., the nonce, or number used to generate a random sequence of numbers) is unique to a given client—as identified by the clientID in the SPA packet—the SPA Message String field can be used to transmit meaningful data by the client. This does not require any further processing, policy evaluation, or the establishment of a TCP or TCP/TLS connection.

An ideal use case for this functionality involves the secure collection of monitoring data from distributed IoT sensors. These edge devices transmit small amounts of telemetry data regularly; embedding this data within a SPA packet allows for it to be securely transmitted without incurring the overhead of establishing a TCP/TLS connection. That said, the recipient (i.e., the AH) must be expecting this data; since the data transmission is unidirectional, the sender receives no verification that the data is actually received (i.e., *fire and forget* SPA packet transmission).

3.3.4 Alternatives to SPA

SPA alternatives can be readily incorporated as part of a system that supports the SDP and ZT principles of Least Privilege and Need-to-Know. For example, an SDP architecture could use a globally accessible enterprise IdP. In this scenario, the IdP would have a control channel to the SDP controllers and accepting hosts. An initiating host successfully authenticating to an IdP would trigger a control plane message informing the SDP controllers and accepting hosts, which in turn expect an immediate incoming connection from the initiating host's IP address. The initiating host would then be allowed to establish a TCP connection to the controller and accepting hosts.

4 Mutual Authentication

SDP architectures require both communicating parties to authenticate against each other, thereby providing inherent security benefits such as mutually authenticated connections, reduced risk of forged certificates, and MITM protection. The following sections discuss the mutual authentication process as well as these key benefits.

The first step is authentication at the SDP controllers, followed by SPA, and then mutual authentication to establish an encrypted connection between distributed SDP components. Mutual authentication works by using mutual transport layer security (mTLS) or a comparable technology (e.g., IPSec or SSL) to allow two sides of a communications channel to verify each other's identity with certificate-based authentication.

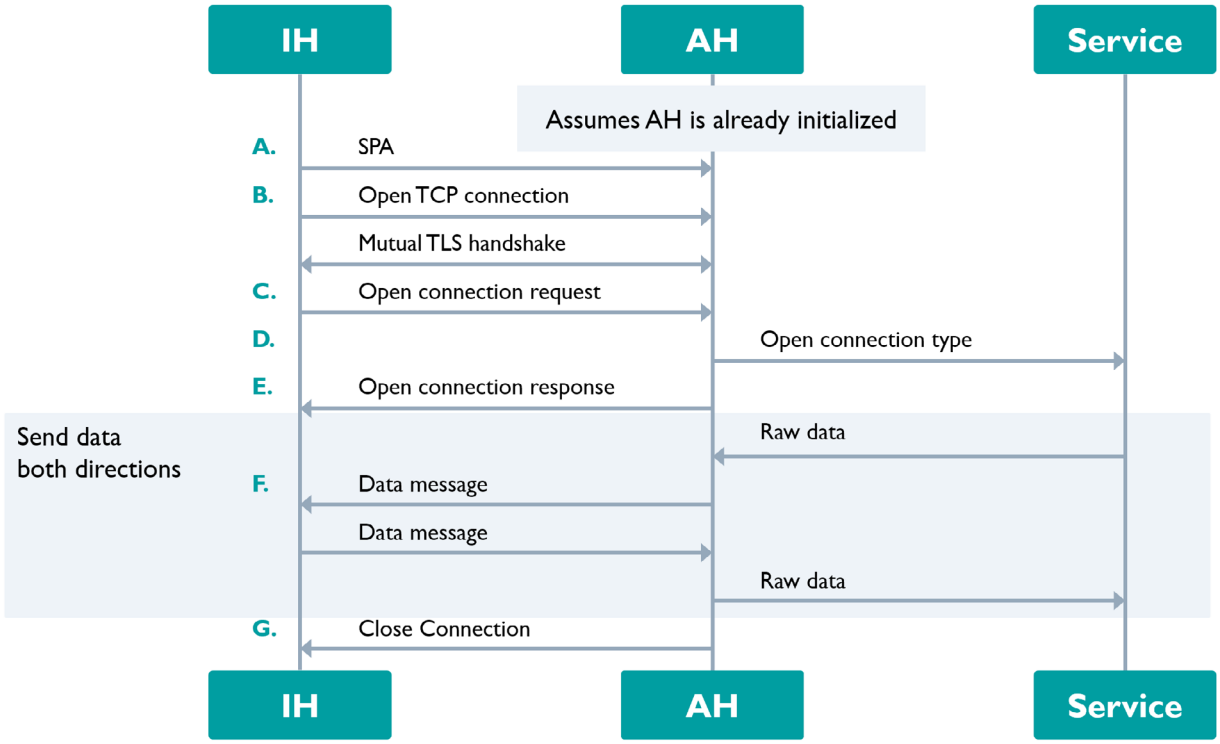

Figure 2: *Sample IH-AH flow*[4]

In the displayed sequence diagram, the IH opens a connection with the AH where a gateway is installed. Initially, a SPA packet is sent to authenticate further communications. Once completed, an mTLS handshake is conducted where public and private keys are evaluated from both the IH and AH—this is to enable non-repudiation, thereby ensuring that the IH and AH are both authentic. Once the mTLS connection is established, data is sent between the IH and AH as needed.

Careful planning will ensure that the SDP architecture's mutual authentication mechanisms are working as expected. Weak cipher suites and solutions that do not support mutual authentication should be avoided, as they lack sufficient security for preventing common attacks. Specifically,

[4] Figure adapted from Cloud Security Alliance, "Software-Defined Perimeter (SDP) Specification v2," 10th, March, 2022, https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-zero-trust-specification-v2/

the connections between the IH and AH and the IH-controller require the use of mTLS, whereas the controller-AH connection should use mTLS as an option. Both TCP/TLS and UDP/DTLS are acceptable for SDP implementations; however, mTLS is an optional feature in TLS that must explicitly be turned on and enabled.

Additionally, the root certificate for SDP components should involve an enterprise public key infrastructure (PKI) system or SDP-specific certificate authority (CA) and not rely on pre-issued or implicitly trusted certificates associated with consumer browsers. Browser technologies are common targets of impersonation attacks in which attackers forge a certificate from a compromised CA. SDPs should include processes for ensuring that certificates can be revoked efficiently using the Online Certificate Status Protocol (OCSP) or comparable mechanisms.

4.1 Mutually Authenticated Connections

SDP requires its distributed components to mutually authenticate with each other to ensure that all connections are properly authenticated and encrypted. Specifically, all SDP components must establish an encrypted tunnel for exchanging digital certificates to authenticate one another. mTLS ensures that the parties at either end of the network connection are who they claim to be through the verification of their respective private keys. Mutually authenticated connections enable strong user and device authentication while reducing the potential for vulnerabilities associated with long-lived access tokens from IdPs (e.g., SAML2 or OpenID), forged certificate use, and MITM attacks.

4.2 Reduced Risk of Forged Certificates

Mutual authentication schemes pin certificates to known valid, trusted root CAs. This makes it difficult to forge certificates, as it would require compromising a root CA. SDP's requirement for mutual authentication also mitigates this risk. SDP architectures should incorporate OCSP to check a certificate's validity/status in real-time—specifically, in enabling authentication services to check with the SSL-certificate issuing CA to validate that the certificate in question has not been revoked.

4.3 Man-In-The-Middle (MITM) Protection

Standard TLS does not require both clients to be authenticated, making it possible for a MITM to impersonate a client. Mutual authentication prevents MITM-based attacks (e.g., MITM TLS protocol downgrade attacks) by requiring both parties to authenticate with each other prior to communicating. This mutual authentication (i.e., the mutual verification of the communicating entities' signed certificates) mitigates the likelihood of a MITM attack by ensuring that each component holds a valid private key issued by a trusted authority. Messages are read only after verifying the sender on each side; sessions are terminated if one party is unable to prove its identity.

5 Tunneling

SDPs incorporate encrypted traffic tunnels to achieve one-to-one isolation between connections. The following sections outline tunneling's key features and benefits—namely application access segmentation and tunnel-confined access.

SDP establishes a separate, secure connection (i.e., tunnel) for each request per authorized/authenticated entity. These separately tunneled layers are long-lived and shared across multiple applications, enabling SDP to act as an encrypted overlay for all IP network types, including traditional on-premises networks, software-defined networks (SDNs), and cloud-based IT infrastructures. By providing seamless integration across various deployment models and service layers, SDP enables organizations to reduce management complexity while normalizing security across heterogeneous environments and simplifying network and security operations.

5.1 Application Access Segmentation

In an SDP architecture, mTLS connections are created between each pair of SDP components—namely, the IH and AH—where each component validates access to the other component when establishing a secure connection. In the case of application layer data, traffic will be routed via the mTLS tunnel established between the user device and the AH. SDP specifically allows multiple connections from an IH to an AH. However, each IH gets its own tunnel to the AH.

5.2 Tunnel-Confined Access

Because users/devices are restricted to their established tunnel-confined access to requested resources, malicious actors and their activities (e.g., lateral movement, privilege escalation) are restricted to the tunnel's boundaries. SDP mandates the containment of an entity's activities to the tunnel—this effectively limits the potential for compromise and protects the organization's network from both insider and external attacks. SDP contains and controls user access within the boundaries of the established tunnel, providing an individual, isolated connection for each application or device between the IH and AH. For application access at layer 7 of the Open Systems Interconnect (OSI) model, individual connections can be protected by a tunnel established between the SDP agent at the IH and the application server's port via the gateway. This design ensures that the user cannot access the application server host machine and discover other users and devices on the network.

6 Micro-Segmentation

The widespread adoption of cloud computing and technologies like virtual machines (VMs), containers, and edge computing have led to expanding attack surfaces and an increase in lateral moving threats. Micro-segmentation mitigates these threats by enforcing security controls and applying access controls/restrictions per isolated micro-segment, thereby enabling organizations to maintain discrete traffic control and monitoring capabilities. This partitioning of the organization's network limits the damage caused by breaches and prevents threat actors from moving laterally across segments.

The following sections discuss the importance of workload-level micro-segmentation, as well as relevant scenarios, approaches, and policy controls.

6.1 Micro-Segmentation of Workloads

The ability to partition segments down to the workload level is crucial for organizations implementing/managing SDP architectures. In contrast, traditional network segmentation only divides the data center network into sub-networks to prevent attackers from moving inside the network perimeter.

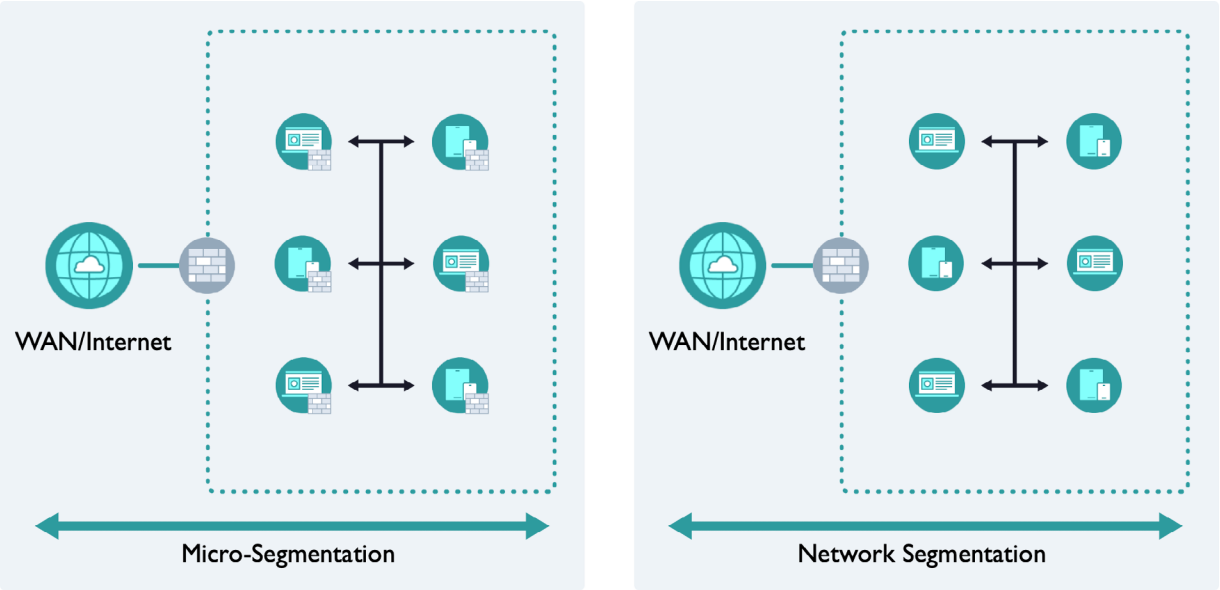

Figure 3: *Workload Segmentation Versus Network Segmentation*[5]

By enabling workload-level micro-segmentation granularity, organizations can protect traffic flows between applications (i.e., east-west, server-to-server) and achieve one or more of the following scenarios:

- VMs can be segmented individually, or as a group.
- Workloads and applications can be segmented based on a single running instance or into a group that comprises all running instances of the application.
- Workloads can also be segmented based on the operating system used, such as a segment consisting of all Windows or Linux instances.

6.2 Micro-Segmentation Scenarios & Approaches

Different micro-segmentation scenarios and approaches are possible depending on the SDP deployment model used. The following table summarizes the level of segmentation achieved through SDP's primary deployment models.

Description	Segmentation Achieved
Client-to-Gateway	This model provides micro-segmentation by securing connections between clients and gateways, but does not micro-segment connections to servers behind gateways. Servers behind the gateway are not segmented and any server compromise can laterally spread to other servers.
Client-to-Server	This model provides network micro-segmentation by securing all connections to servers. In addition, servers hosting gateways are hidden.

[5] Figure adapted from Rashmi Bhardwaj, Network Interview, "Micro segmentation vs Network Segmentation"

Server-to-Server	This model provides network micro-segmentation by securing all connections to the server. In addition, servers hosting the gateway are hidden. To achieve true workload-level micro-segmentation, server-to-server SDP should be implemented.
Client-to-Server-to-Client	This model achieves workload-level micro-segmentation. Client only sees relevant workloads.
Client-to-Gateway-to-Client	Workloads are not relevant in this scenario.
Gateway-to-Gateway	This model provides micro-segmentation by securing connections between gateways, but does not micro-segment connections to servers behind gateways.

Table 2: *Micro-Segmentation Scenarios & Approaches*

As mentioned previously, the server-to-server and client-to-server-to-client SDP deployment models support workload-level micro-segmentation approaches. Additionally, organizations can take the following approaches to further segment their environments for optimal ZT.

- **Application-based micro-segmentation:** High-value applications running on bare metal servers, VMs, or containers can be protected by creating a micro-segment and restricting east-west communications from outside the segment. This is an ideal method for meeting compliance regulations and the requirements of oversight bodies/authorities.

- **Environment-based micro-segmentation:** Separates environments into isolated segments (e.g., development, testing, and production) — this segmentation type cannot be achieved by traditional measures, because environments may be physically spread across multiple on-premises data centers and/or in the cloud.

- **Tier-based micro-segmentation:** Segments and isolates each application tier (e.g., web server (or, front-end), application server (or, back-end), database (or, data tier) to prevent malicious actors from moving across the overall application stack. This style of isolation creates depth in the defense by tucking in the data tier behind the app and web tier. The attacker has to jump across two tiers to get to the data.

- **Process-based micro-segmentation:** Highly granular segmentation that operates at the process or service level; for example, a specific software service is isolated and only allowed to communicate on specific network paths, protocols, and ports. Process-based segmentation is even more fine-grained than the broad stroked three tier-based segmentation.

- **User-based micro-segmentation:** This is classic zero trust network access (ZTNA) scenario, where users only have access to applications based on the principle of need to know. Even though the server-to-server communication is open (rather, out of scope), there is no risk of threats coming from user laptops and getting access to privileged applications.

- **Network-based micro-segmentation:** Traditional environments containing operational technologies (OT) and industrial control systems (ICS) may require network-based micro-segmentation in order to implement an organization-wide SDP architecture. This is because OT/ICS devices are usually incapable of supporting SDP client software installation. In these cases, network-based micro-segmentation is implemented using network devices as enforcement points and relies on subnets, virtual local area networks (VLANs), or similar mechanisms to create segments. Policies can then be configured and enforced per SDP and applied to subnets or VLANs, versus individual hosts.

6.3 Micro-Segmentation Policy

Policy enforcement controls should in theory allow organizations to permit authorized applications and entities to connect and communicate with each other, while at the same time enabling them to meet unique governance, security, and compliance requirements. In practice, numerous challenges—particularly those brought on by cloud/hybrid infrastructures—can make creating and enforcing flexible policies more difficult. Some of these challenges include:

- Choosing the right granularity level and scope for creating micro-segmentation policies
- Reducing the organization's attack surface and increasing resilience to withstand/protect against security breaches
- Remaining agile and adaptable despite having more granular policies
- Managing policies for thousands of workloads over varied locations
- Creating unified policies across multiple cloud architectures and service providers
- Enforcing security controls at both the network and process level for true defense-in-depth

These micro-segmentation challenges can be addressed by using a centralized policy plane that prevents intrusions from spreading laterally while allowing security teams to modify policies quickly in response to new/evolving security use cases or changing business requirements. Apart from enforcing segmentation policies, SDP security controls should also continuously monitor drifts in access patterns and flag policy violations to ensure that the organization maintains a resilient security posture as applications, systems, and environments evolve over time.

7 Identity & Access Management (IAM)

Identity and access management (IAM) systems verify identities while enabling organizations to store managed attributes and group memberships related to those identities. Within an SDP architecture, SDP can be used to authenticate users, inform SDP authorization decisions, and enrich audit logs with access information about users from registered devices. SDP's access controls are typically based on factors like IAM attributes as well as the attributes of the connecting devices. The combination of user and device criteria helps in creating granular access rules that only allow connections from authorized users and devices to specific applications.

By focusing on application access and users, versus network access and IP addresses, IAM is capable of yielding more useful connection details for logging, while at the same time significantly reducing IT management overhead. SDP relies on the identity attributes and group memberships managed by IAM tools: as user attributes or group memberships change, SDP automatically detects these changes and modifies a given user's access.

SDP architectures are designed to integrate with existing enterprise IAM solutions and technologies (e.g., OAuth2, SAML2, LDAP, Active Directory), as well as continuous step-up authentication measures like MFA or prompting for a OTP under certain circumstances (e.g., to access sensitive systems or connect over remote access). IAM systems can also communicate with SDP via API calls in response to identity lifecycle actions such as the disabling of accounts, changes to group memberships, and dropping user/device connections.

The following sections discuss the various approaches to implementing IAM as well as its underlying technologies and protocols.

7.1 IAM Approaches

The implementation of robust IAM is vital for organizations looking to adopt SDP. With proper IAM controls in place, risk can be assessed continuously and contextually, and more granular permissions granted, all in line with SDP requirements. The following IAM approaches—namely, role-based access control (RBAC), attribute-based access control (ABAC), and policy-based access control (PBAC)—are commonly used when implementing an SDP architecture, and are discussed in the following sections.

7.1.1 RBAC

RBAC is an access control and management approach that leverages roles, users, and role assignments.

- **Roles** are created within the access management system and are assigned access attributes to specific resources. Typically, these roles correspond to a specific job function. For example, a role might be assigned the appropriate access attributes for adding journal entries in the organization's financial systems.

  Users represent human or system entities that are assigned unique identifiers (e.g., user IDs). To securely implement RBAC, user authentication is a critical step in the process. User authentication is typically governed by an IAM system which verifies a user's identity via passwords, certificates, one-time-passwords, and other related means.

- **Role assignments** tie together the role and the user. By assigning a user to a role, the user is granted the specific access governed by that role. Likewise, removing a user from a role also removes the user's access governed by that role. The addition/removal of the user to/from a role can be manually performed by a human, which presents a risk of untimeliness and lack of completeness, especially during the access removal process. The addition/removal can also be automated using ties to another system. One example is automated role assignment based on placement in an org chart (e.g. managers in the HR department might be automatically granted read access to employee feedback results).

Depending on the complexity of the systems involved, RBAC can take a significant effort to design and implement. Defining roles and related permissions can be a lengthy and very technical process. In addition, role permissions need to be reviewed periodically (e.g., annually) to validate access rights. User/role assignments also need to be reviewed periodically in totality, and whenever a user changes roles or leaves the company. In the long run, successful RBAC implementations typically see an overall reduction in user management effort.

While not as flexible or as granular as ABAC[6] or PBAC, RBAC is the most prevalent IAM approach, as most organizations have leveraged RBAC for decades and have invested a great deal of time and effort in their RBAC framework. These RBAC frameworks can be leveraged by SDP implementations to grant access to specific application and system functionality.

7.1.2 ABAC

A derivative of RBAC, ABAC leverages six key components: attributes, subjects, objects, operations, policies, and environmental conditions.

- **Attributes** are characteristics of a subject, object, or environment conditions. Attributes contain information given by name-value pairs.
- **Subjects** are human users or non-people entities like devices that issue access requests to perform operations on objects. Subjects are assigned one or more attributes (e.g., ID, job roles, group memberships, departmental/organizational memberships, management level, security clearance) that define their access rights.
- **Objects** are system resources that contain/receive information, for which access is managed by the ABAC system (e.g., devices, files, records). It can be the resource or requested entity, as well as anything upon which an operation may be performed by a subject (e.g., data, applications, services).
- **Operations** are the execution of a function upon an object (e.g., read, write, edit, delete), at the request of a subject.
- **Policies** are the representation of rules/relationships that make it possible to determine if a requested access should be allowed, given the values of the attributes of the subject, object, and possibly environment conditions.
- **Environmental conditions** define an operational or situational context in which access requests occur. Environment conditions are detectable environmental characteristics. Environment characteristics are independent of subject or object, and may include the current time, day of the week, location of a user, or the current threat level.

ABAC can provide more granular access decisions based on attributes of the requesting entity, the object to which access is being requested, the intended use of the object and the environment conditions (e.g, geolocation, network) or request context. This granular access control complements SDP's fine-grained security model. That said, ABAC can be complicated to implement, since it requires the definition of a potentially vast number of attributes to establish rules and policies around.

7.1.3 PBAC

PBAC builds upon ABAC by leveraging digital policies containing a combination of logical rules to dynamically control (e.g., allow/deny) a human's or system's access to a specific resource. PBAC policies are composed of both attributes and roles to determine access rights, with theoretical privileges compared to actual privileges and differences automatically applied.

[6] NIST, "Guide to Attribute Based Access Control (ABAC) Definition and Considerations," January, 2014, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-162.pdf

For example, a role defined for a manager may require the attachment of specific types of accounts on the single sign-on server, web server, and database management system. Appropriate users are then attached to the role. Like ABAC and RBAC, PBAC can also be leveraged in an SDP implementation to provide an additional layer of identity-based security.

7.2 IAM Technologies & Protocols

The following IAM technologies and protocols—namely, Federated Identity Management (e.g., IdP), Security Assertion Markup Language 2.0 (SAML2), and OpenID Connect/OAuth 2.0—allow organizations to implement a robust SDP architecture focused on user identity and specific risk posture. For example, the appropriate IAM controls can be used for determining whether authentication requirements should be tightened based on the current conditions, engagement, transaction, or nature of the requested resource.

7.2.1 Federated Identity Management

Federated identity management involves the use of an IdP to make assertions regarding an entity's risk posture. At its heart, a series of cryptographic operations build the trust relationship and exchange credentials. A practical example is a user logging in to their company's network, which hosts a directory server for accounts. The user then opens a browser connection to a SaaS application. Instead of logging in, there are a series of behind-the-scenes operations, where the IdP (e.g., the internal directory server) asserts the identity of the user, and that the user is authenticated, along with any attributes. In an SDP architecture, the SDP controller would rely on assertions from the IdP to permit access to the application.

7.2.2 Security Access Markup Language (SAML)

Security Access Markup Language 2.0 (SAML2) is an open standard that enables an IdP to pass authorization information to cloud service providers, thereby allowing users to leverage one set of credentials for many website logins. SAML2 greatly reduces login and management complexity, and when used in an SDP architecture, fulfills requirements for continuous authorization and authentication. For this reason, the standard is highly preferred by organizations looking to create a federated security access model for sending security information (e.g., identities and access privileges) to a service provider in a secure, standardized manner.

7.2.3 OpenID Connect & OAuth 2.0

OpenID Connect is an identity layer positioned on top of the OAuth 2.0 protocol that allows for user identity validation based on the authentication performed by an authorization server. An SDP architecture can integrate OpenID Connect to leverage a framework of IdPs providing user authentication as a service. Since OpenID Connect was designed to encourage an open ecosystem, IdPs can include both large entities (e.g., Google, Microsoft, Apple) as well as smaller organizations that provide their own IdP services. Organizations looking to integrate OpenID Connect and OAuth 2.0 into an SDP implementation can either select a specific IdP already available or create their own organization-specific OpenID Connect mechanism.

7.2.4 Cross-Domain Identity Management Systems

System for Cross-Domain Identity Management (SCIM) is an open standard for exchanging identity information between domains and facilitating secure communications between cloud-based applications. Designed for strong security and interoperability, SCIM can be used for provisioning and deprovisioning accounts in external systems and for exchanging attribute information. Like the other IdP technologies covered in this section, SCIM can be leveraged in an SDP implementation to provide identity-centric management and customized resource access/traffic policies for cloud-based applications.

8 Secure Remote Access

Traditional perimeter-centric network models were designed in an era when strong security entailed allowing on-premises (i.e., inside the perimeter) users access to data and applications, while denying access to everyone else. As remote users started requiring secure access to protected data and applications from outside the perimeter, VPN technologies were introduced to securely extend the organization's boundary. Unfortunately, VPN authentication and encryption methods can be easily intercepted and bypassed; subsequently, SDP-based secure remote access is widely considered an ideal replacement for VPN-based solutions.

The following sections delve into VPN's critical shortcomings, as well as SDP's benefits as an ideal alternative to VPN for secure remote access.

8.1 Issues with Traditional VPNs

Although VPNs were initially helpful in addressing secure remote access requirements, they are not without their own security gaps and related operational issues. By extending the organization's boundary to remote users' endpoints, the IT environment's footprint increases significantly—and with it, the corresponding resource requirements to manage the extended coverage. Additionally, the risks increased exponentially as each endpoint required greater management and security sustainment, substantially increasing the enterprise attack surface. Furthermore, VPNs only solidified the notion that there was an implicitly trusted interior where anything inside of the enterprise boundary, including remote endpoints, were automatically trusted. Overall, legacy VPN architectures expose the enterprise to attacks and adversely impact the user experience, especially when accessing cloud applications.

8.2 SDP for Secure Remote Access

When adopting a ZTA, SDP addresses many of the operational and security shortcomings introduced by VPNs. As SDPs are based on software rather than hardware, they are location- and infrastructure-agnostic allowing them to be deployed anywhere to protect either or both on-premises and cloud infrastructures. SDPs are largely different from VPNs due to direct endpoint-to-resource connections rather than providing wide network access to users connecting over VPNs. Additionally, while VPNs primarily rely on physical hardware appliances such as firewalls or VPN concentrators deployed at

specific physical locations, SDP enables the reduction in latency, capacity constraints, and complexity through a software-native architecture that scales to support secure access to protected resources regardless of where it resides. SDP also integrates with modern, secure technologies such as mTLS encryption, MFA, and dynamic access policies. Overall, SDP provides superior secure remote access capabilities in comparison to VPN while also enabling ZTA adoption.

9 Software-Defined Networking (SDN) & SDP

Because SDP is often mistakenly conflated with SDN, it's worth comparing and contrasting these two approaches to infrastructure design. SDN is an approach to networking that enables dynamically configured, highly efficient and easy-to-manage networks. Despite sharing similar attributes such as separate data and control planes and policy-defined controls, SDN and SDP are different infrastructure frameworks that ultimately serve different purposes; that said, they can work in conjunction with one another to deliver optimal ZT.

The following sections provide an overview of SDN, some similarities it shares with SDP, and how the two differ.

9.1 Software-Defined Networking

SDN's approach to networking involves the automating of network provisioning, management, and control via centralized software-driven policies for managing network endpoints. This management of network services through the abstraction of higher-level functionality renders the network more flexible, dynamic, and easier to manage. Like SDP, SDN works by decoupling the control plane from the data plane. However, in the case of SDN, planar separation is primarily for improving speed, efficiency, and throughput in routing network traffic versus improving security and preventing cyber attacks. With SDN, network operators can build networks that are more agile and adaptable and flexible in their configurations—for example, a service provider may wish to allocate more network bandwidth to business applications during the day, shift the same bandwidth to general Internet applications in the evening, and then prioritize bandwidth for batch and backup applications late at night.

A typical representation of SDN architecture consists of three layers: the application layer, control layer, and infrastructure layer.

- **Application layer** contains applications that explicitly and programmatically communicate the desired network behavior and network requirements to the control layer.
- **Control layer** receives service requests from the application layer and requests the infrastructure layer to execute them, thereby dynamically optimizing the workload of the infrastructure layer.
- **Infrastructure layer** contains the routing equipment and networking devices responsible for handling packets, based on the rules offered by the control layer.

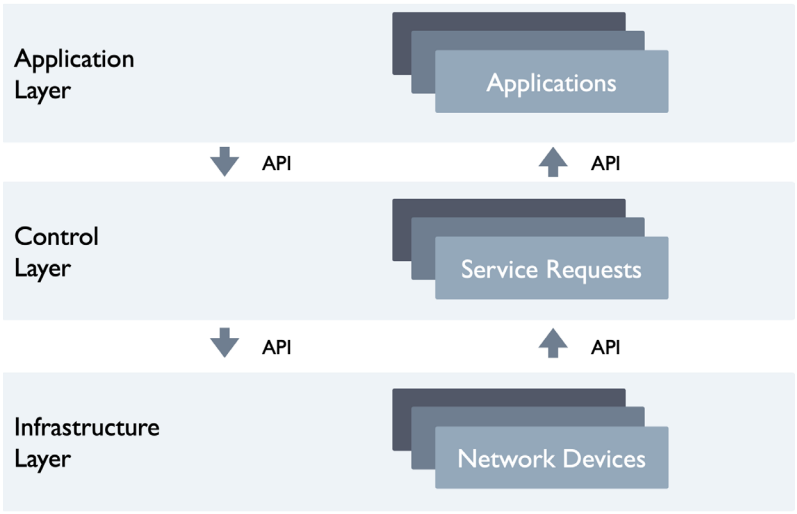

Figure 4: *Interactions Between Different SDN Layers*

Because SDN is an open architecture with its three layers communicating via open API standards, multiple vendor products can be implemented at each layer, thereby ensuring the highest degree of interoperability between key industry players' technologies and solutions.

9.2 SDN Versus SDP

SDP utilizes the same data plane, control plane, and application or orchestration plane framework as SDN. The data plane involves the actual data passed within the mTLS tunnel between the client and application resources. The control plane involves the SPA for access requests between IH and AH or between IH and controller for onboarding. The application or orchestration plane involves communication between the controller and IdP. In essence, the SDP is an implementation of SDN for enabling ZTNA.

Whereas SDN is primarily geared for network optimization and control centralization through the establishment of a dynamic networking infrastructure, SDP is designed to protect and secure IT assets and resources. In the case of SDP, the limiting of access to authenticated and authorized users enables secure connections at every network layer, based on pre-established policies set by the organization. SDP can also be used to optimize the security of an SDN-based network infrastructure—for example, by embedding SDP components into the SDN environment to protect a set of services connected to the network. In the same vein, SDP can just as easily be deployed in conventional, legacy networks to achieve ZT.

Conclusion

In this module, we covered some of the challenges and shortcomings of traditional security architectures and discussed how SDP architectures address these challenges. We then covered the features and capabilities of SDP that make it ideal for implementing a ZT architecture in today's IT environments, as well as the key features and technologies that together constitute an SDP architecture. Lastly, we compared and contrasted SDP and SDN, highlighting their similarities like data and control plane separation while illustrating their differing use cases.

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