软件定义边界(SDP)标准规范2.0

cloud security alliance®

CSA GCR GREATER CHINA REGION cloud security alliance®

软件定义边界工作组的官网地址是:

https://cloudsecurityalliance.org/research/working-groups/software-defined-perimeter-and-zero-trust/

序言

为深入贯彻落实国家关于建设网络强国、数字中国、智慧社会的战略部署，在各行业主管部门的监督和政策指引下，信息化水平快速提升，构筑全方位的网络安全体系也成为网络安全核心任务。

自 2019 年工信部发布的《关于促进网络安全产业发展的指导意见(征求意见稿)》文件中零信任安全首次被列入“网络安全需要突破的关键技术”以来，零信任 SDP 技术架构被行业广泛探索和采用。随着各行各业应用实践的不断深入， SDP 的标准规范也应随之发展。此次发布的 SDP 2.0 标准规范充分考虑了这些年零信任 SDP 的技术发展和行业需求，更好地满足数字化安全的未来发展。

此次发布的 SDP2.0 标准规范是一次国际协作的成果，CSA 大中华区零信任工作组组长陈本峰与其他国际上的零信任安全专家共同编写了该标准规范，我国的国密算法也作为推荐的加密算法首次进入 SDP 标准规范。

SDP 为网络运营者提供动态灵活的边界功能部署能力，聚焦于保护关键的组织资产，可实现精准授权，降低网络攻击的可能性。SDP 是零信任原则不可分割的一部分，它帮助零信任安全实现最小授权原则，隐蔽网络和资源。

信息安全是动态发展的，新技术不断推陈出新，SDP 就是在传统的技术生态中迭代形成的一套技术架构，它是多种网络安全技术的整合，包括密码技术、网络技术、访问控制技术和软件开发技术等，SDP 适用场景也非常广泛，包括云计算、物联网、大数据、工业互联网、移动互联网等，为 SDP 体系架构的发展提供了更多的可能。

在“后疫情时代”的背景下，网络资源快速开发和利用，远程办公、线上教育、勒索病毒、网络攻击、网络诈骗等对我们管理能力提出了严峻挑战，面对网络环境的复杂变化，SDP2.0 体系的应用将为网络空间的健康发展起到重要的支撑作用。

本规范通过通俗易懂的语言向大家介绍了 SDP2.0 的体系架构、部署模型、访问流程等，希望你在读完本规范后可对 SDP 有更为清晰的理解，并帮助你更快完成应用实践。

李雨航 Yale Li

CSA 大中华区主席兼研究院院长

致谢

本文档《软件定义边界（SDP）标准规范 2.0》(Software-Defined Perimeter (SDP) Specification v2.0)由 CSA 软件定义边界工作组专家编写，CSA 大中华区秘书处组织翻译并审校。

中文版翻译专家（排名不分先后）：

组　长：陈本峰

翻译组：单美晨　何国锋　何伊圣　贺志生　黄　超　梁小毅　林冠烨
刘洪森　穆域博　秦益飞　苏泰泉　汪　海　王　彪　王贵宗
王　亮　许木娣　姚　凯　于新宇　余　强　余晓光

审校组：陈本峰　高　巍　郭鹏程　姚　凯

研究协调员：周建金

感谢以下单位对本文档的支持与贡献：

云深互联(北京)科技有限公司　　中国信息通信研究院
北京奇虎科技有限公司　　腾讯云计算（北京）有限责任公司
华为技术有限公司　　启明星辰信息技术集团股份有限公司
深信服科技股份有限公司　　安易科技（北京）有限公司
中国电信研究院安全技术研究所　　北京山石网科信息技术有限公司
北京天融信网络安全技术有限公司　　上海安几科技有限公司
北京中宇万通科技股份有限公司　　北京北森云计算股份有限公司
江苏易安联网络技术有限公司　　北京赛虎网络空间安全技术发展有限公司

英文版本编写专家

主　编：

Jason Garbis　　Juanita Koilpillai

参编作者：

Junaid Islam　Bob FloresDaniel Bailey　　Benfeng Chen （陈本峰）

Eitan Bremler　Michael Roza　Ahmed Refaey Hussein

特别鸣谢：

Larry Hughes

评审人：

Alistair Cockeram　　Takahiro Ono　T Prasad　　Nya Murray　Michael Rash

CSA 分析师：

Shamun Mahmud

CSA 全球工作人员：

Claire Lehnert

1.0 版参编作者：

Brent Bilger　Alan Boehme　Bob Flores　Zvi Guterman　Mark Hoover

Michaela Iorga　Junaid Islam　Marc Kolenko　Juanita Koilpillai

Gabor Lengyel　Gram Ludlow　Ted Schroeder　Jeff Schweitzer

CSA 分析师

软件定义边界 (SDP) 和零信任工作组是国际云安全联盟 (CSA) 研究工作组，倡导并促进零信任安全理念的采用。该工作组就如何将零信任理念应用于云和非云环境提供了实用且合理的指导。该工作组将以NIST零信任研究成果和方法论为基础并加以利用，推荐 SDP 作为零信任安全理念应用的基本架构。该工作组将不断获取和编纂基于行业经验的丰富知识，并修订和扩展 SDP 规范。最后，SDP 工作组认识到对 SDP 采取包容性方法的必要性。 因此，该工作组支持替代架构，只要这些架构符合零信任原则。

在此感谢以上专家。如译文有不妥当之处，敬请读者联系CSA GCR秘书处给与雅正！联系邮箱：research@c-csa.cn；云安全联盟CSA公众号。

献辞

谨以本文纪念胡安妮塔-库尔皮拉（Juanita Koilpillai），她是一位安全领域的领导者、影响者、导师和朋友。胡安妮塔是所在领域的杰出人物，在成为两家成功初创企业的创始人和首席执行官的道路上突破了诸多无形障碍。Juanita 将她的知识分享给从事技术工作的年轻女性作为指导，以回报社区。Juanita 在 IEEE 女性组织(IEEE WIE)中发挥了重要的指导作用，还创建了一个慈善基金纪念她的父亲，并创立了 MERGE 这一支持当地青年项目的非营利组织。Juanita 在撰写此书以及其他 CSA、IEEE 和 NIST 出版物方面发挥了重要作用。

目录

1. 介绍

软件定义边界(SDP)架构提供了动态灵活的网络安全边界部署能力，以在不安全网络上对应用和服务进行隔离。SDP 提供了隔离的、按需的和动态配置的可信逻辑层，缓解来自企业内外部的网络攻击。SDP 对未经授权实体进行资产隐藏，建立信任后才允许连接，并通过单独的控制平面和数据平面管理整个系统。 企业借助 SDP，可以实现零信任安全的目标，并且建立有效性和弹性的安全体系，从而摆脱传统（且基本无效）的基于物理边界防御的模型。

1.1 意义

该规范是对国际云安全联盟 CSA 的软件定义边界工作组 (SDP WG) 于 2014 年 4 月发布的《软件定义边界 (SDP) 标准规范 V1.0》（以下简称“SDP v1”）的升级。

尽管初版的规范是完整的，但没有充分讨论组件加载流程、保护 NPE 非人类实体（Non-Person Entities）等方面的问题[1]。 此外，自 SDP v1 发布以来，SDP 架构得到业界广泛认可，并包含在我们现在所说的零信任理念中。相比初版，本次修订版本的 SDP 标准规范进行了扩展和加强（包含对内容的添加、澄清和延展），并反映了当前最新的零信任行业状态。

值得一提的是，该修订版基于 SDP 工作组发布的 SDP v1 及后期发布的其他文档进行修订，特别是基于《SDP 术语表》和《SDP 架构指南》。 这两个文档的链接在本文的“参考文献”章节。

1.2 范围

该标准规范包含了 SDP 的架构组件、交互流程和基础安全通信协议，重点关注控制平面如何在安全边界内授权安全连接，以及数据平面如何在在发起主机 IH 和接受主机 AH（服务器、设备、服务）之间实现安全连接。

1.3 读者

本标准规范的目标受众是：

- 在企业中部署零信任和 SDP 产品的解决方案架构师和安全主管
- 在方案中使用 SDP 架构实施零信任安全理念的供应商和技术提供商

1 非人类实体（NPE）是一个具有数字身份的实体，在网络空间中行动，但不是人类行为者。这可以包括硬件设备、软件应用和信息制品。

2. SDP 设计

SDP 的目标是让企业安全架构师、网络提供商和应用程序所有者能够:

- 部署动态的“软件定义”边界
- 隐藏网络和资源
- 防止非授权访问企业的服务
- 实施以身份为中心的访问策略模型

SDP 将物理的安全设备替换为安全逻辑组件，无论组件部署在何处，都在企业的控制之下，从而最大程度地收缩逻辑边界。SDP 执行零信任原则，即强制执行最小特权访问、假设被入侵、以及“信任但验证”，仅在认证和身份验证成功后，基于策略来授权对资源的访问。

SDP 的设计初衷是为 IPv4 和 IPv6 网络提供有效且易于集成的安全架构，包括对控制平面组件的保护和访问控制。SDP 为跨数据平面通信的机密性和完整性提供保障，还包括一个“需知访问“模型，要在经过设备验证以及身份认证（用户和非人类实体 NPE）成功之后才能以加密方式登录到边界网络。

SDP 的设计理念上提供多个层次的无缝集成 – 包括用户、用户设备、网络和设备的安全。SDP 适用于任何基于 IP 的基础设施，无论是基于硬件的传统网络、软件定义网络 (SDN)，还是基于云计算的基础设施。SDP 的双向验证隧道实际上是一个加密层，可以部署在任何一种 IP 网络之上。因此，SDP 能将多个异构的环境统一成通用的安全层，从而简化了网络、安全和运维。

对于云计算基础设施，SDP 在 OSI 网络模型七层中的五层集成了安全性，即：

- **网络层：**在该层以虚拟化[2]的方式提供计算、存储和监测。
- **传输层：**在该层云API将虚拟化资产与资源池和用户联系起来。
- **会话层：**该层用于管理底层虚拟化基础设施。
- **表示层：**该层用中间件来管理应用层和应用。
- **应用层：**为用户提供商业价值。

未集成 SDP 的 OSI 层是数据链路层和物理层，TCP/IP 网络模型将这两层合并为网络层。

有关 SDP 和分层网络模型的更多信息，请参见附录 B。

作为 SDN 和网络功能虚拟化(NFV)的补充，SDP 可以保护 SDN 创建的基于 IP 的网络连接。

基于过往的反馈以及《SDP 标准规范 1.0》实施中的经验教训，本次更新的《SDP 标准规范 2.0》将进一步阐明上一版本标准规范中定义的以及《SDP 架构指南》白皮书中阐述的各种部署模型中的访问控制问题[3]。

SDP 提供了一个显著更好的方法预防、监测和应对那些针对应用程序和基础设施的各种网络

2 参见关于 SDP 和网络功能虚拟化（NFV）的论文 - https://www.waverleylabs.com/resources/publications/

3 https://cloudsecurityalliance.org/artifacts/sdp-architecture-guide-v2/

攻击及跨域攻击。SDP 通过尽可能缩小攻击面、采用最小特权原则实现这一点，即使用户通过不受信任的公共网络（如互联网）访问资源也能得到保护。

传统的网络安全解决方案侧重于保护网络和系统的安全，而 SDP 侧重于以身份为中心保护数字资产。从传统边界防护转变到 SDP，使企业能够更加从容地应对 DDoS、凭证失窃和对企业资源的勒索软件等攻击。

2.1 SDP 概念

SDP 聚焦于保护组织机构的关键资源，而非组织机构的边界。它能够为网络的所有层面定义和执行基于风险的、动态的，以身份为中心的、且上下文感知的访问策略。SDP 为定义和执行访问策略提供了基础，这些策略对于业务、应用程序和网络的负责人来说意义重大，尤其在组织机构内第一次实施的时候。

SDP 能够为应用程序和企业资源所有者提供的边界防护能力有：

- 可以将服务安全部署到假定被入侵的网络上（即“假定被入侵”）。
- 通过不受信任的网络访问企业资源时，可以精细化调整用户身份权限。一个典型的用例是替换VPN。

SDP 使用由应用负责人控制操作的逻辑组件取代了传统边界防御设备（通常是物理的）。SDP 通过访问策略进行设备认证和身份验证后，才允许用户对应用程序的访问。

SDP 背后的原理并非全新的。美国国防部（DoD）和美国情报机构（IC）内的多个组织，已经实施构建了相应的网络架构，即访问网络之前先进行身份验证和授权。通常在机密或“高端”网络（如国防部定义的网络）中，所有服务器都隐藏在远程访问网关设备后面，用户必须完成身份验证，才能被授予服务的可见权限并开放访问通道。SDP 借鉴了机密网络中使用的逻辑模型，将其合并进入标准工作流程。多年来，相关安全负责人逐步达成了对这些概念的共识，最具代表性的开端就是 2004 年举办的 Jericho 论坛。近期，在美国国家标准与技术研究院（NIST）中定义的零信任架构中也包含了这些原则[4]。

SDP 保留了上述“需知”（最小特权）模型的优点，同时克服了必须借助远程访问网关设备的不足。事实上，SDP 访问控制设计的初衷是面向所有用户，而不仅仅是远程用户。SDP 要求任何终端在获得对受保护的服务器和相关服务的网络访问权之前，首先要进行终端的身份验证和鉴权，然后就在请求系统和应用程序之间实时创建加密连接。概括来说，SDP 可以在对相关资源（如用户、设备和服务）完成安全验证后，允许其在一个特定边界中访问所需的服务，这些服务对未经授权的资源保持不可见。

2.2 SDP 的架构和组件

简单来说，SDP 由两个逻辑组件构成：SDP 主机和 SDP 控制器。

4 –参见 NIST 零信任架构文档 – SP 800-207 https://csrc.nist.gov/publications/detail/sp/800-207/final

SDP 主机，通常是全栈主机或轻量级服务，可以发起或接受连接。这些动作由 SDP 控制器管理，通过控制平面上的安全信道交互。数据则通过数据平面中单独的安全信道通信。控制平面与数据平面分离，实现系统架构的灵活性且高度可扩展性。此外，出于规模化或可用性的目的，所有组件均可以做冗余部署。

SDP 主机（发起主机或接受主机）与 SDP 控制器进行通信，SDP 控制器是一个设备或服务器进程，它确保用户经过身份验证和授权、以及设备得到验证，并建立安全通信，保证网络上的数据流量和控制流量是分离的。

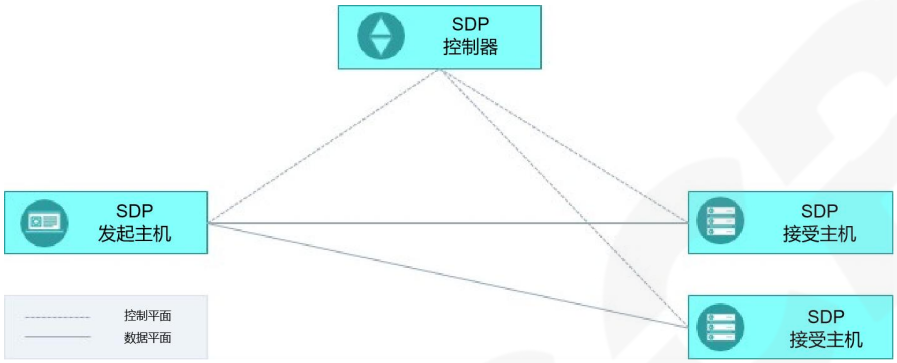


图 1:SDP 架构（之前由 CSA 在《软件定义的边界和零信任》中发布）[5]

SDP 的架构由以下组件组成：

- **SDP控制器**—该组件的设计初衷是用于管理所有的身份验证和访问流程。SDP控制器本质上是整个解决方案的“大脑”，负责定义和评估相应访问策略。它充当了零信任架构下的策略决策点（PDP职能）。SDP控制器负责同企业身份验证方（例如，身份提供商IdP、多因子身份验证MFA服务）的通信，统一协调身份验证和授权分发。它是一个中心控制点，用于查看和审计所有被访问策略定义的合法连接。
- **发起主机（IH）**-这类访问实体可以是用户设备或NPE（非人类实体），例如，硬件（如终端用户设备或服务器）、网络设备（用于网络连接）、软件应用程序和服务等。SDP用户可以使用SDP客户端或浏览器来发起SDP连接。
- **接受主机（AH）**-这些实体是逻辑组件，通常被放置在受SDP保护的应用程序、服务和资源的前端。AH充当零信任架构下的策略执行点（PEP职能）。PEP通常由具备SDP功能的软件或硬件实现。它根据SDP控制器的指令来执行网络流量是否允许发送到目标服务（可能是应用程序、轻量级服务或资源）。从逻辑上讲，AH可以与目标服务可以部署在一起或者分布在不同网络上。

这些 SDP 组件可以部署在本地或云上，出于扩容或可用性目的可以进行冗余部署。

以下我们详细介绍每个组件：

1）SDP 控制器

SDP 控制器是一个关于策略的定义、验证和决策的组件（零信任架构中的策略决策点 PDP），

5 https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-and-zero-trust/ 第 6 页

其维护的信息包括：哪些身份（如用户和组）可以通过哪些设备访问组织架构中的服务（本地或云中）。它决定了哪些 SDP 主机可以相互通信。

一旦用户（在 IH 上）连接到控制器，控制器将对该用户进行身份验证，并根据用户的上下文（包括身份和设备属性）判定是否允许其访问被授权的服务（通过 AHs）。

为了对用户进行身份验证，控制器可以使用内部用户表或者连接到第三方的身份和访问管理（IAM）服务（本地或云中）执行认证，并且可以加上多因子认证（MFA）。身份验证方式通常基于不同用户类型和身份。例如，企业员工可以通过身份验证提供商进行身份验证，而外部承包商可以通过存储在数据库中的凭据或使用联合身份进行身份验证。

为了对用户访问服务进行授权，控制器可以使用内部的“用户到服务”映射策略模型，或第三方服务：如 LDAP、活动目录（AD）或其他授权解决方案（本地或云上的）。授权通常由用户角色和细粒度信息决定：基于用户或设备属性，或者用户被授权访问的实际数据元素/数据流。实际上，SDP 控制器所维护的访问控制策略可以由其他组织型的数据结构（如企业服务目录和标识存储）来输入。通过这种方式，SDP 控制器实现了 NIST 定义的零信任原则中的动态零信任策略。

此外，控制器可以从外部服务获取信息，例如地理位置信息或主机验证服务，以进一步验证（在 IH 上的）用户。此外，控制器可以向其他网络组件提供上下文信息，例如有关用户身份验证失败或访问敏感服务的信息。

SDP 控制器与零信任 PDP 概念组件密切相关。根据 SDP 架构的配置需求，它可以部署在云上或本地。

SDP 控制器由单包授权（SPA）协议的隔离机制保护，使其对未授权的用户和设备不可见和不可达。该机制可以由控制器前端的 SDP 网关提供，也可以由控制器本身提供。

2）SDP 发起主机 IH

SDP 的发起主机 IHs 与 SDP 控制器通信，以便开启通过 AH 接受主机来访问受保护的公司资源的过程。

控制器通常要求 IH 在认证阶段提供用户身份、硬件或软件清单以及设备健康状况等信息。控制器还必须为 IH 提供某种机制（如凭证密钥），以便 IH 与 AH 建立安全通信。

IH 的形式可以是安装在终端用户机器上的客户端程序或 Web 浏览器。使用客户端程序可以提供更丰富的能力，例如主机检查（设备安全状态检查）、流量路由和更便捷的身份验证。

发起主机（IH）最重要的作用之一是使用 SPA 启动连接，本文稍后将对此进行详细讨论。在某些实现中，SPA 报文可能由基于浏览器的 SDP 客户端生成。IH 可以是人类用户的设备（如员工或承包商的计算机或移动设备）、应用程序（如胖客户端）或是物联网（IoT）设备（如远程水表）。在刚刚的最后一个例子（远程水表）中，其身份是一个非人类身份，但还是需要经过身份验证和授权。有关这个话题的更多讨论，请参阅*软件定义边界与物联网设备*章节。

3）SDP 接受主机 AH

接受主机（AH）是 SDP 策略执行点（PEP），用于隐藏企业资源（或服务）以及实施基于身份的访问控制。AH 可以位于本地、私有云、公共云等各种环境中。

受 AH 保护的服务不仅限于 Web 应用程序；可以是任何基于 TCP-或 UDP 的应用程序，例如 SSH、RDP、SFTP、SMB 或胖客户端访问的专有应用程序。

默认情况下，对 AH 的任何网络访问都被阻止，只有经过身份验证和授权的实体才能访问。

如上所述，AH 可以与目标服务部署在一起，或者分布在不同的网络上。这些模型如图 2 所示。

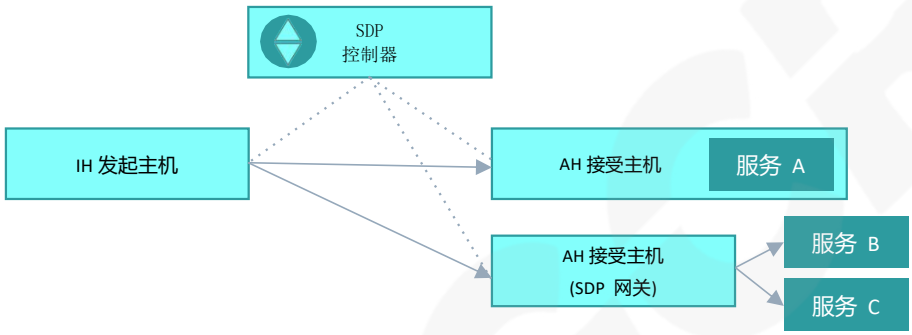


图 2

接受主机（AH）从 SDP 控制器接收控制信息，并只接受经过控制器确认的那些发起主机（IH）的连接请求。利用从控制器接收到的控制信息，接受主机（AH）保证只有经过授权的发起主机（IH，包括用户和设备）才能访问到受保护的服务。

接受主机（AH）作为安全通信的交换站，从发起主机（IH）接收访问流量然后转发到被保护的后端服务。后端服务的响应信息通过接受主机（AH）返回到发起主机（IH）。

SDP 是一个面向连接的逻辑层协议，可以用来保护多种网络拓扑架构。下图 3 介绍了多种 SDP 部署模型和架构的细节。它们在《SDP 架构指南》一文中有详细的说明，阐述了不同部署模型下各个 SDP 组件的配置。

2.3 SDP 部署模型

SDP 把客户端（包括人类和非人类实体）连接到资源（在下图中资源被描述为服务器）。资源可以是任意类型能被网络访问到的服务。它可以是在物理服务器中或虚拟机中运行的服务，或是在 IaaS、PaaS 平台上运行的服务或容器化的服务。

本节概述了六种 SDP 部署模型。尽管不同模型使用了不同的网络拓扑，但在逻辑上提供了同样的价值，都是对受保护的资源进行严格的访问限制。要请参阅《SDP 架构指南》[6]了解这些部署模型的详细情况。

6 关于完整的描述，请参见 CSA 软件定义边界架构指南, 2019 年 5 月，第 14 到 18 页

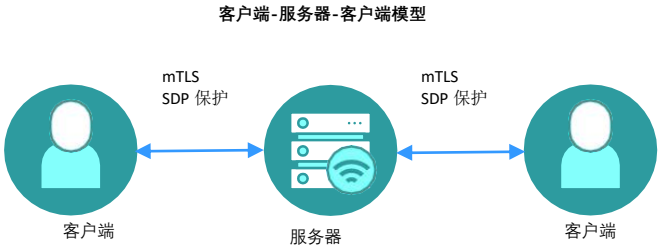
下图中，蓝线表示被双向认证加密协议（如 mTLS 协议和 IKE 协议等）保护的网络连接，从而可以抵御中间人攻击（MITM）。灰线表示使用应用程序原有协议的网络连接，这些连接可能是加密的，也可能是未加密的。在图表中为了表述简单，省略了 SDP 控制器。

其中有部分部署模型用到了 SDP 网关。SDP 网关是 SDP 接受主机（AH）的软件程序，为受保护的后端服务提供隔离性和访问控制能力。

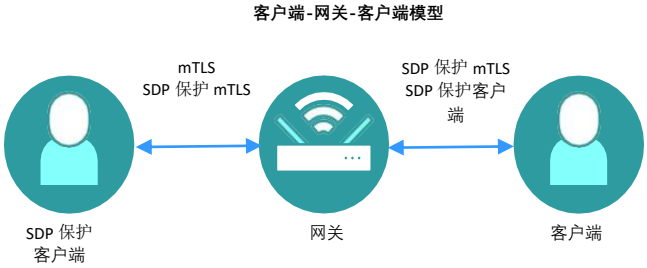
当一个或多个服务器需要被 SDP 网关保护时，不论底层网络拓扑如何，发起主机（IH）和接受主机（AH，网关）之间的通信连接将会被加密保护。在客户端-网关模型中，网关可以被远程访问，但同时也是隐藏的，从而提供了安全边界。这个模型不需要受保护服务器侧进行任何改造。

当一个组织机构需要端到端的安全通信的时候，客户端-服务器模型将服务器和接受主机（AH）绑定在同一个主机上。在这个模型中，服务器隐藏在安全边界内的，服务器端必须要安装一个 SDP 软件，用来实现客户接受主机（AH）的功能。

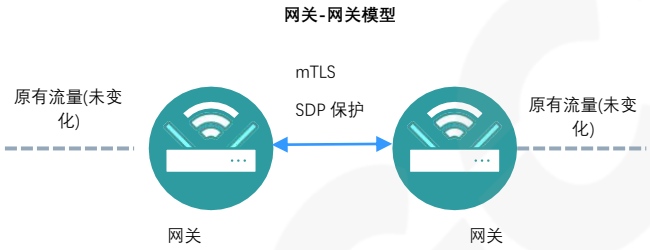
服务器-服务器模型下，不论底层网络拓扑如何，可以保证所有服务器之间的通信连接全部经过加密来保护。在这个模型下，服务器都隐藏在安全边界内。



在一些点对点的通信场景中（如 VOIP、聊天和视频会议服务等），点对点的流量都通过中间服务器来转发。在这个模型下，服务器隐藏在安全边界内。



客户端-网关-客户端模型是客户端-服务器-客户端模型的变体。这个模型支持在点对点网络协议下，由一个客户端直接向其他客户端连接的场景中执行访问控制策略。在这个模型下，网关都隐藏在安全边界内。



在 SDP 规范 1.0 中未包含网关-网关模型。这个模型很适合用于某些物联网场景。在这个模型下，网关都隐藏在安全边界内。

图 3 – SDP 部署模型

2.4 SDP 工作流程

一般来说，SDP 组件工作流程流通常分为两种独立的类型：加载（Onboarding）流程（每个组件均有独立流程）和访问流程（在多个组件之间协调）。就定义而言，每个 SDP 组件都有一个单独的加载（Onboarding）流程，并参与多个访问流程。

以下所述的工作流程只是代表一般情况，因为在不同的 SDP 实现方案和不同的 SDP 部署模型之间的细节会有所不同。

1）控制器加载（Onboarding）流程

每个 SDP 系统都需要一个或多个控制器。为了让加载流程得以成功，至少必须保证有一个控制器在任何时候都是可用的。一些 SDP 实现方案中，需要一个始终在线的控制器使访问流程获得成功。控制器必须是可从其他任何 SDP 组件的运行位置进行网络访问。因此，它们通常是可通过互联网在全球范围内可达，但仅限于获得授权的用户/设备。

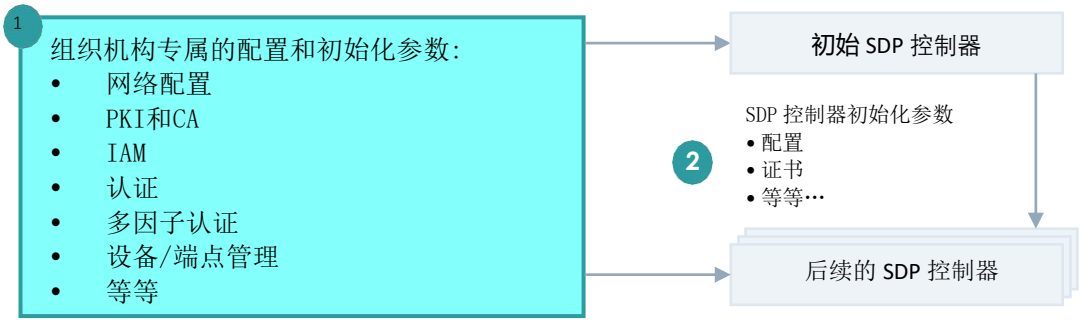


图 4 – 控制器加载流程

控制器的工作流如图 4 所示：一个初始的（主要的）SDP 控制器被引入服务，并连接到适当的、可选的身份验证和授权服务（例如，PKI 颁发证书机构服务、设备身份验证、地理位置、SAML、OpenID、OAuth、LDAP、Kerberos、多因子身份验证和其他此类服务）。SDP 控制器应该时刻在线，以便对任何其他 SDP 组件随时可用。如有必要，也可以让后续更多的控制器上线，并使用相同的组织机构的专属配置，以及来自初始控制器的初始化参数（配置）信息加载。

为了实现负载均衡和冗余，许多 SDP 的实现方案会支持控制器集群部署。任何 SDP 实现方案都必须支持这样一种机制：后续的控制器可以加入集群，连接到集群内的其他控制器，并共享或访问任何当前的状态信息。这种机制依赖于具体实现方案，本文不进行详细讨论。

2）接受主机（AH）加载流程

每个 SDP 系统都需要一个或多个接受主机 AH。它们可以使用上述的任何 SDP 部署模型进行部署。也就是说，它们可以是独立的网关，也可以作为服务器（资源/业务系统）的一部分部署。

AH 可以是长期在线的，也可以是短暂的——两者在 SDP 实施中都是可以接受的。独立网关 AH 可能寿命较长，运行数月或数年。但也可能是短暂的，如在基于负载进行扩展或收缩的动态网关集群中。

部署在单个服务器（业务系统）中的 AH 在线时间可长可短。在这种场景下，它们的生命周期将与它们所属的服务器实例的生命周期绑定在一起。服务器实例可以是长期存在的，例如传统的 Web 或应用服务器；也可以是短期存在的，如 DevOps 基础设施的一部分服务。

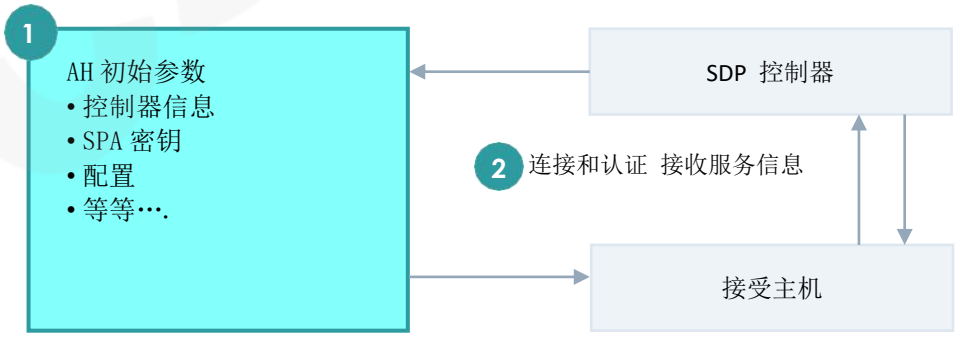


图 5-接受主机（AH）加载流程

AH 加载流程如图 5 所示：当 AH 投入使用时，它们必须连接到 SDP 系统中的一个或多个控制器并进行认证。一旦加载成功，它们就可以接收 SPA 报文，并处理来自授权 IH 的访问。

任何 SDP 实现方案都必须支持这样一种机制：所有 AH 都可以被配置为连接到控制器集群中。由于这种机制依赖于具体实现方案，因此具体内容不在本文的讨论范围内。同样地，许多 SDP 实现方案都支持 AH 集群部署，以实现负载均衡和冗余容灾的目的，这是一种比较常见的网关部署模型。

3）发起主机（IH）加载流程

发起主机（IH）可以是用户设备、或非用户操作的系统（如物联网设备或充当 IH 的服务器）。与 SDP 系统中的其他所有组件一样，IH 也需要加载，如图 6 所示。在这个流程中，它们首先需要被配置连接到控制器所需的初始参数信息：包括网络信息（主机名或 IP 地址），以及任何必要的共享密钥（例如，SPA 密钥、数字证书）。由于这种机制依赖于具体实现方案，因此具体内容不在本文的讨论范围内。通常，IH 加载流程需要使用到企业身份管理服务供应商，并且在用户设备上需要做用户身份认证。

IHs 只需要加载一次，之后就可以启动访问流程。

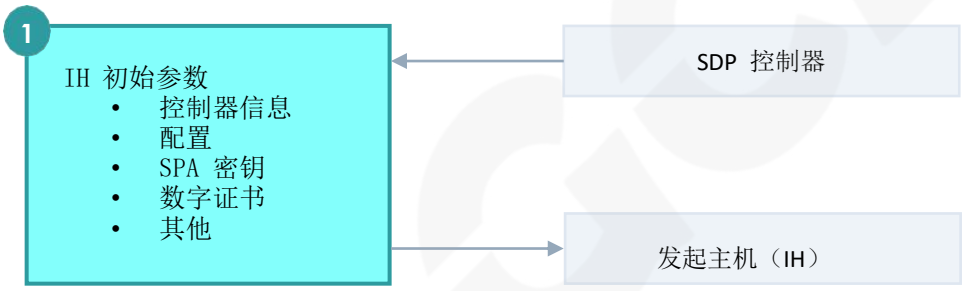


图 6 – 发起主机(IH)加载流程

4）访问流程

发起主机 IH 启动访问流程以连接被 SDP 系统保护的业务系统（资源），参考表 1 和图 7。访问流程涉及了 SDP 全部组件：发起主机 IH，控制器，接受主机 AH。

步骤	访问流程
1	当已加入的 IH 重新上线时（例如，在设备重新启动后，或当用户启动连接时），它将连接到 SDP 控制器进行身份验证
2	在 IH 身份验证（在某些情况下，使用其相应的身份提供商）成功后，SDP 控制器会确定该 IH 有权（通过 AH）通信的服务列表。
	但此时控制器尚未将此列表发送给 IH；必须等到步骤 3 之后。
3	SDP 控制器指示 AH 可以接受来自 IH 的通信，以及向 AH 发送用于建立用户、设备和服务之间双向加密通信所需的信息。
4	SDP 控制器向 IH 提供已授权的 AH 和服务列表，以及建立双向加密通信所需的任何可选信息。
5	IH 使用 SPA 协议发起与授权 AH 的连接，首先验证 SPA 中的信息以确保安全，然后 IH 建立与 AH 之间的双向 TLS（mTLS）加密连接。

表 1：访问流程

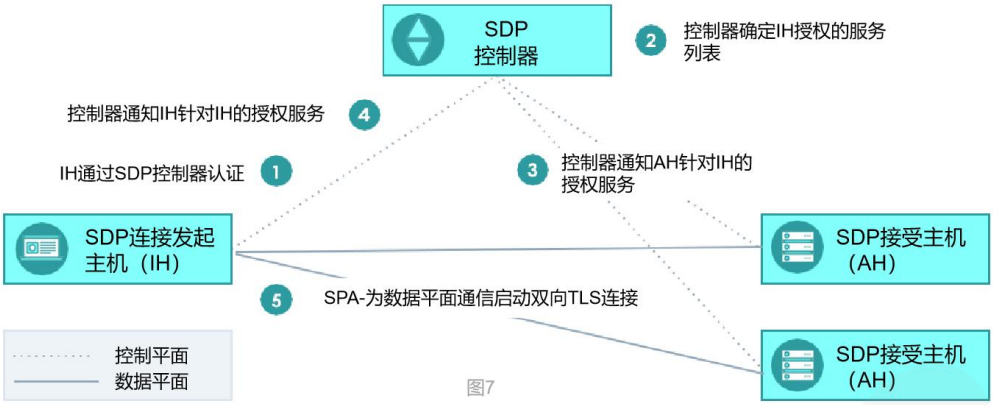


图7

2.5 IH 加载流程示例

SDP 控制器、IH、AH 和用户的加载，根据图 2 所示的部署模型以及具体实现方案的不同情况会有差异。SDP 系统的创建和运行可以通过应用程序编程接口（API）或管理界面进行。

如上述流程所述，当 SDP 系统部署和配置完成之后， 接下来 SDP 控制器和 AH 上线。

以下时序图是使用了外部身份提供商（IdP）进行身份验证的 IH 用户加载流程[7]。

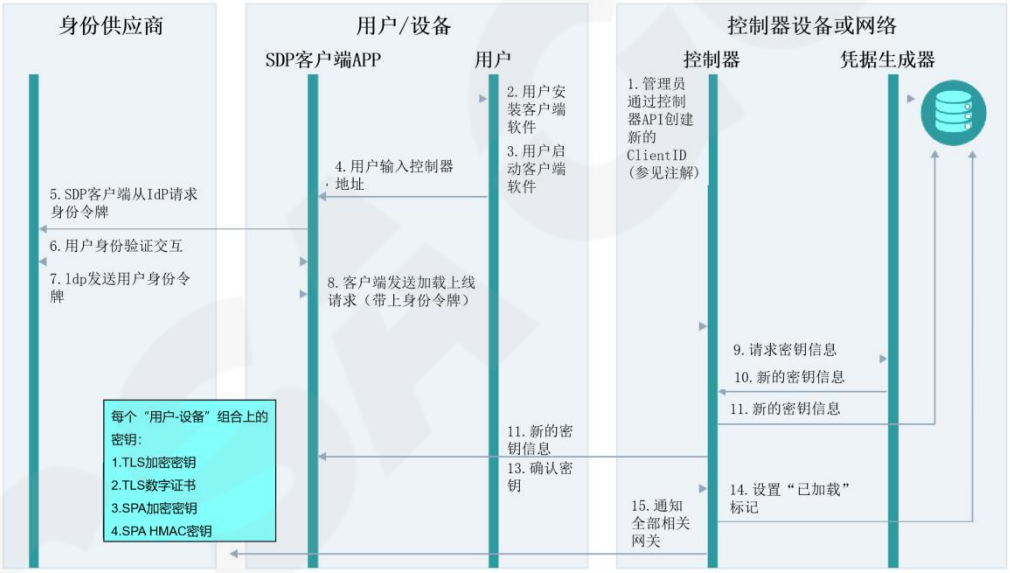


图 8 — 加载（IdP 场景）

注解：此示例使用身份提供商（IdP）的信息用于用户加载上线。加载系统可以包括多因子认证功能，例如 MFA 和设备属性验证(如企业部署的数字证书或终端管理软件)。此示例还省略了向各节点分发密钥信息(如 SPA 密钥或其他共享密钥)的步骤，该步骤和 SDP 的实现方案有关。例如，开源 SDP 实现方案中会生成此密钥信息，然后依赖企业的部署方式将其安全地传输给 SDP 的组件。

在该示例工作流程中，IH 使用客户端(上图中的 SDP 客户端)与身份提供商(IdP)协商。值得注意的是，对于某些 SDP 产品模型，SDP 控制器将与 IdP 建立起信任关系（例如，验证客户端转发

7 基于 SDP 的开源参考实现 - https://www.waverleylabs.com/open-source-sdp/

给它的 SAML 令牌）。

2.6 单包授权(SPA)

SDP 的核心原则之一是，任何未经授权的实体不仅不能访问业务系统（资源），还不得访问 SDP 基础设施本身。为达到这一目标，要求实体与任何 SDP 组件连接之前必须要通过基于密码学的授权验证。这种机制提高了 SDP 的安全性和弹性：未经授权的实体无法与 SDP 组件建立网络连接，因此无法尝试漏洞利用、无法尝试暴力破解或利用被盗的用户账密。这与传统的远程访问解决方案（如 VPN）形成了鲜明的对比，后者往往暴露在互联网上的所有恶意攻击者面前。[8]

SDP 中实现这个目标的机制就是单包授权(SPA)。SDP 版的 SPA 是基于 RFC 4226 HMAC 的一次性密码“HOTP”[9]，后面会阐述 SPA 包中如何使用它。

值的一提的是，虽然 SDP 标准规范始终围绕着如何使用 SPA 保护对控制器和 AH 的访问，但也存在其他合理架构下的可靠替代方案。请参阅后面的 SPA 替代方案章节部分。

SPA 在 SDP 中实现的主要原则如下：

> •**隐藏 SDP 系统组件：**控制器和 AH 不会对来自远程系统的任何连接尝试作出反应，直到它们提供了对该 SDP 系统合法可信的 SPA 报文。具体地说，在基于 UDP 的 SPA 的情况下，主机不会响应 TCP SYN，从而避免了向潜在攻击者泄露任何信息。(具体实现示例：配置了“默认丢弃”规则的防火墙)。无论是独立的 AH(SDP 网关)，还是逻辑上属于服务器/业务系统一部分的 AH，都采用这样的原则。
>
> •**减轻对 TLS 的拒绝服务攻击：**运行 HTTPS(使用了 TLS)协议的面向互联网的服务器非常容易受到拒绝服务(DoS)或分布式拒绝服务(DDoS)攻击。SPA 可以减轻这些攻击，因为它可以让服务器在产生建立 TCP 或 TLS 连接的开销之前快速拒绝未经授权的连接尝试，而同时允许授权连接的建立，即使 DoS 攻击还在进行中。
>
> •**攻击检测：**从任何其他主机发往控制器或 AH 的第一个报文必须是 SPA 报文。如果 AH 收到任何其他报文，则应将其视为攻击。因此，SPA 让 SDP 能够根据单个恶意报文确定攻击。

值得一提的是，对于 SPA 报文进行验证的计算开销应该是比较轻量的，从而提高 SDP 系统抵御 DDoS 攻击的恢复能力(如在国际云安全联盟 SDP 工作组的《SDP 作为 DDoS 防御机制》白皮书中所述[10])。

SDP 组件之间的通信由 SPA 发起：IH-控制器、AH-控制器和 IH-AH。SPA 报文使用 UDP 或 TCP 协议，具体取决于所选的实现方案。

基于 UDP 的 SPA 为受 SPA 保护的服务器提供上述所有安全优势。与基于 UDP 的 SPA 相比，基于 TCP 的 SPA 只具备其中一部分优势。具体地说，使用 TCP SPA 的 SDP 组件将向所有远程(和潜在的恶意)用户暴露开放的端口，因此服务器不会被隐藏。该服务器也将部分受到 DDoS 攻击--它允许从任何远程 IP 地址建立一个 TCP 连接，然后在建立 TLS 连接之前执行 SPA 验证。TCP 连接

8 公平地说，有一些商业和开源的 VPN 类解决方案确实隐藏了基础设施，例如 Wireguard 就是这样一个开源案例
9 详见 https://tools.ietf.org/html/rfc4226
10 https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-as-a-ddos-prevention-mechanism/

比 TLS 连接的资源消耗要小得多，但它确实也会消耗服务器资源，使服务器面临一定程度的 DDoS 风险。总的来说，使用 TCP SPA，TCP 可以让服务器通过识别无效 SPA 包来鉴别攻击，但这种识别只有在建立 TCP 连接之后才能做到，因此会消耗服务器资源。

关于最小化服务暴露面，另一个值得注意事项是 AH 和控制器的 DNS 枚举攻击。直接通过 IP 地址或仅通过私有 DNS 服务进行连接可能会降低服务对攻击者的可见性。也就是说，SDP 服务组件的公有 DNS 解析（例如 controller1.sdp.mycompany.com）的本身可能就对恶意攻击者暴露了攻击面（例如：攻击者可以对 SDP 基础设施的进行大规模 DDoS 攻击）。

请注意，为了提高 SPA 协议的安全性和弹性，下文中推荐的 SPA 消息格式相比 SDP 标准规范 v1 版做了改进和更新。

1）SPA 消息格式

虽然不同 SDP 实现方案的 SPA 消息格式可能不同，但所有 SDP 系统都应支持 SPA 作为在组件之间启动连接的机制。值得一提的是，SPA 报文创建者和接收者应当具有共享的信任根，因为每个 SPA 报文都需要共享密钥才能构建有效的 SPA 报文。建立信任根（即如何将共享密钥安全地传递到 SDP 组件）取决于具体实现方案，超出了本文的讨论范围。通常来说，这些信息包括在 IH 和 AH 的加载流程中。

用户 ID	为每对用户–设备分配 256 位数字标识符。该字段用于区分发送报文的用户、设备或逻辑组。
非重复随机数	16 位随机数据字段，用于避免 SPA 报文被重复使用，以防止重放攻击。
时间戳	通过确保 SPA 报文在短时间内的有效性(例如 15 到 30 秒)，以防止处理过期无效的 SPA 报文。这也提供了一种机制减少接收方所需的重放攻击检测缓存。
源 IP 地址	发起主机的公开可见 IP 地址。接受主机不依赖报文头中的源 IP 地址，因为在传输过程中很容易修改。IH 必须能够获得 IP 地址，供 AH 使用，作为报文的来源地址。
消息类型	该字段是可选的。它可用于通知接收者在建立连接后， IH 会发送什么类型的消息。
消息内容字符串	此字段是可选的。它将取决于“消息类型”字段。例如，这个字段可以用于标明 IH 将请求的服务（如果在连接时已知目标服务）。
HOTP（基于 HMAC 的一次性密码)	这个一次性哈希密码是基于 RFC 4226 所描述的算法以及共享密钥生成的。SPA 报文中必须要使用 OTP 以验证其真实可靠性，其他 替代的 OTP 算法也可以应用在这里，只要能达到验证 SPA 报文真实可行性的总体目标。
HMAC 值	基于上述所有字段计算得出。算法可选择 SHA256(推荐)、SHA384、SHA512、SM3、Equihash 或其他高效鲁棒的算法。HMAC 使用共享(密钥)种子计算。将此 SPA 消息的所有先前字段合并然后计算 HMAC 值，最后 AH 会使用它来验证 SPA 消息的完整性。HMAC 验证在计算上是轻量级的，因此可以用来提供抵御 DoS 攻击的弹性能力。任何带有无效 HMAC 的 SPA 报文将被立即丢弃。

表 2：SPA 消息结构

值得一提的是，其他 SPA 实现方案可能包含额外的加密方式，例如，使用 IH 的私钥(以实现不可假冒)，或 AH 的公钥(以实现保密)。然而，非对称加密在计算上开销比较大，建议只有在轻量级验证机制(如简单的 HMAC)通过之后才应该被接收方使用，以保持 AH 对 DoS 攻击的弹性。

2）SPA 作为一种安全的、独立的、无连接的消息传输协议

在 SDP 系统中，SPA 的一个有趣的“副产品”用例是，SPA 包不仅可以用作发起连接的手段，还可以用作从远程对象传输数据的手段。因为 SPA 报文基于一个共享的密钥，接收方可以相信其中包含的数据是由一个有效的 SDP 客户端发布的。

如果 SPA 种子密钥对于特定的客户端是唯一的(由 SPA 报文中的 ClientID 标识)，那么消息内容字符串字段可以被客户端用来传输有意义的数据。这既不需要任何进一步的处理或策略评估，也不需要建立 TCP 或 TLS 连接。

这个方案对一组需要定期传输少量数据的分布式物联网传感器很有用。将数据嵌入到 SPA 报文中，从而使这些设备完成数据传输任务而避免产生 TCP/TLS 连接的额外开销。当然，这种机制也有一些缺点。接收方(AH)必须随时待命接收这个数据，而且因为这是一个单向传输协议，即便数据通过“发送后就不管”的 SPA 报文传输被收到，发送方也不会收到确认消息。尽管如此，在某些数据并非关键且事关重大的场景下，SPA 数据传输可以是一种有用的方案。

3）SPA 的替代方案

SDP 架构在设计时考虑到了灵活性，因此这次对标准规范的修订中包含了各种可能的方案和部署模型。例如，SDP 定义了六种部署模型，每种部署模型都可以用来解决不同的场景，以不同的方式实现，但有不同的取舍。SPA 是一种健全和安全的机制，可以落地实现 SDP 的关键原则，即隐藏基础设施，提供了对 DDoS 攻击的弹性应对能力，并提高了检测攻击的能力。SPA 还具有使 SDP 成为一个闭环系统的优势：一旦种子密钥被分发后，IH 就可以安全可靠地与控制器和 AH 建立加密通信，而无需依赖任何外部系统。

然而，SPA 并不是实现这些目标的唯一机制。有一些替代方案也可以实现上述定义的原则，因此它们可以成为支持 SDP 和零信任原则的系统的一部分。例如，一个无 SPA 的 SDP 系统可以利用一个全球可访问的企业身份提供商，并有一个控制通道通往 SDP 的控制器和 AH。在这种模式下，IH 向身份提供商的认证请求的将触发一个控制面消息给 SDP 控制器和 AH，通知他们 IH 认证成功，并告知它们立即会收到来自 IH 的公网 IP 地址的连接。随即，IH 将能够建立一个与控制器和 AH 的 TCP 连接，因为控制器和 AH 已经预期到这个连接。当然，TCP 连接之后紧跟着会有双向认证的 TLS 连接（mTLS），以及下文将阐述的其他安全层。

当然，每个系统和架构都有自己的设计考量和取舍。只要该方案实现了上面提到的三个原则，基于 SPA 替代方案的系统就可以成为一个健全和有价值的 SDP 系统的组成部分。

2.7 组件之间的传输层双向认证

SDP 在建立连接的过程需要进行多层次的安全校验。第一步是 SPA，如上所述。下一步，也

就是本章节的主题，要求双向认证，它是作为分布式 SDP 组件之间建立安全加密连接的关键步骤。其他更多的步骤，包括设备和用户安全校验，将在后面讨论。

当 SDP 组件利用 SPA 进行安全校验和授权后，SDP 系统中主要组件之间的连接必须使用 TLS 或其他替代方案（如网络密钥交换 IKE），并通过双向身份认证，来确认设备作为 SDP 的授权成员。具体而言，发起主机和接受主机（IH-AH）、发起主机到控制器（IH 控制器）以及控制器到 AH 之间的连接必须使用双向认证。请注意，TCP 和 UDP 都支持 TLS（在 UDP 情况下，它被称为数据报传输层安全（Datagram Transport Layer Security，简称 DTLS）。这两种都适用于 SDP 系统。

不支持双向认证的弱密码软件套件或协议是不可取的，也是不安全的。高安全强度的密码和协议确保每个组件都包含一个由可信机构颁发的有效私钥，通过双向验证数字签名证书，显著降低了中间人（MITM）攻击[11]的可能性。

这些组件的根证书必须是企业 PKI 系统或 SDP 专用 CA。它不能依赖预装的或与用户浏览器关联的默认信任的数字证书，因为这些证书有可能会受到假冒攻击，攻击者因此可以伪造来自被假冒的证书颁发机构的数字证书。SDP 应实施一种方案，以确保被吊销的证书可以有效地被检测到（例如，使用在线证书状态协议【OCSP】[12]或其他机制）。

无论 SDP 系统使用哪种传输层方案，它都必须能够抵御潜在的攻击，如 MITM TLS 降级协议攻击（使用 mTLS 可以避免这种攻击）。

2.8 设备校验

上个章节阐述的传输层双向认证方案可以证明 SDP 的访问设备拥有一个未过期且未被吊销的私钥，但不校验设备是否满足安全或配置要求。

设备校验的目的是证明设备符合安全要求。特别需要注意的是，在企业环境中，控制器被假定为是受信任的设备，因为它们存在于最受控制的环境中。如果 SDP 网关也在企业的控制下，则也假定它们是受信任的组件。也就是说，这些组件在受控和托管环境中运行，这些环境必须受到企业变更管理控制流程和部署流程的约束，并且必须在 SDP 系统中被谨慎地启动上线。因此，对于控制器和网关组件来说，设备验证被认为是可选项。

另一方面，用户设备（IH）需要设备校验。设备校验可缓解凭证盗窃和由此产生的假冒攻击。[13]用户设备只有在成功满足设备校验和总体安全策略的要求后，才被允许连接到 SDP 控制器。此过程确保攻击者将无法访问 AH 上或其背后的服务，即使攻击者拥有用于 mTLS 身份验证的正确私钥。

无论从任何设备传入的报文，只要没有通过 SDP 的验证和授权的，都会被丢弃，因此该过程可以阻止来自未授权设备的任何传入报文。关于设备校验协议和具体细节，其依赖于企业和产品的具体实现方案，超出了本文的讨论范围，但是在未来的文档中会关注。

11 https://wott.io/blog/tutorials/2019/09/09/what-is-mtls
12 https://tools.ietf.org/html/rfc6960
13 私钥和单包验证应安全储存，并定期轮换。这是未来工作的一个关注领域。

SDP 系统必须具备与企业设备管理/端点管理系统进行集成的能力，并将其设备安全态势检查能力纳入设备校验过程。此外，SDP 系统应对 SDP 客户端软件所运行的用户设备本地环境进行设备安全态势检查。例如，SDP 客户端可以验证用户设备上是否包含企业颁发的有效证书，该证书可用于 mTLS 身份验证。或者，它可以验证设备上是否运行了经企业批准的防病毒组件。值得一提的是，这些因素与总体的零信任 SDP 设计方案有关：是否使用设备信息作为访问策略决策制定的附加因素。此外，还值得一提的是，这里的所谓的“设备”也可以指服务器，服务器可以并且应该以与用户设备采用大致相同的方式进行校验。

2.9 软件定义边界 SDP 与物联网设备

NIST 800-207 文档中的有两个核心零信任原则：系统必须确保“所有数据源和计算服务”以及“所有通信”的安全访问，[14]因此零信任安全必须包括物联网设备。广义上物联网 IoT 的概念，不仅包括在大多数 IT 环境中存在的设备（如打印机、IP 摄像头和 VOIP 电话），而且还包含更专业的设备，如环境或医疗传感器或工业控制系统。

如果这些设备是开放的，并且可以支持安装任意完整功能的软件以及相关操作，类似正常的 SDP 发起主机，那么它们可以被视为与典型的 IH 主机没有区别。我们在这里着重讨论以下三类设备之一：1）它们是封闭（不可扩展）设备，企业无法安装其他软件; 2）它们是通用计算机（如 Arduinos），但其计算/内存/存储容量有限；3）或者它们是功能齐全的通用系统，企业选择不在其上部署 SDP 客户端软件。在以上所有情况下，这些都是联网设备，SDP 架构和策略模型中应包括对它们的访问以及它们发起的访问。

为了被 SDP 系统涵盖，不可扩展设备需要一个“上游”网络设备（“代理”或“连接器”）来捕获物联网设备网络流量，并将其流量代理到 SDP 系统中。SDP 代理代表物联网设备充当逻辑 IH。需要注意到是，该代理应该能够完全控制进出物联网设备的流量。

容量有限的设备可以在其上运行 SDP 软件的某些定制版本，但不能也不需要充当全功能 IH。SDP 架构对来自这些设备和可能来自这些设备的其他通信模型保持开放兼容。例如一组低功耗、低容量的环境传感器阵列可以运行一个轻量级 SDP 客户端，该客户端只生成有效的 SPA 报文来安全地传输数据，既不需要建立 TLS 连接，也不需要认证，以一种非常轻量级的方式安全[15]传输数据。

以上这些想法是未来围绕零信任和物联网相关研究工作的良好起点，我们对于在这一领域有更多的研究和成果发布满怀期待。

2.10 访问策略

零信任的核心理念是通过访问策略对受保护资源进行访问控制 (回顾一下零信任的关键概念：策略决策点 PDP 和策略执行点 PEP)。NIST 零信任架构文档的一个核心原则明确指出了：“对资源的访问由动态策略决定，其中包括客户身份、应用程序/服务和请求资产的可观察状态，还

14 参照 NIST 特别出版物 800-207，零信任架构，第六页
15 这提供了消息完整性，但不保证隐私，因为数据在传输中未加密（尽管它可以用单独发布的密钥进行加密）

可能包括其他行为和环境属性[16]。”网络安全与基础设施安全局(CISA)零信任成熟度模型讨论了策略如何从成熟度传统级别的静态策略，转变到高级级别的跨域输入和输出，再到最优级别的“基于自动/观察触发器的动态策略”[17]。

但是，考虑到零信任架构和企业需求的多样性，很难定义一个有效的、广泛适用的策略模型框架和相关词汇表。例如，仅 SDP 就有六种不同的部署模型，每种模型的安全能力略有不同。这种多样性造成的结果是，早期的零信任文档往往对策略这个话题闭口不谈。例如，SDP 标准规范的第一版没有讨论或定义策略模型。较新的零信任模型为我们提供了一些如何处理策略的指引。

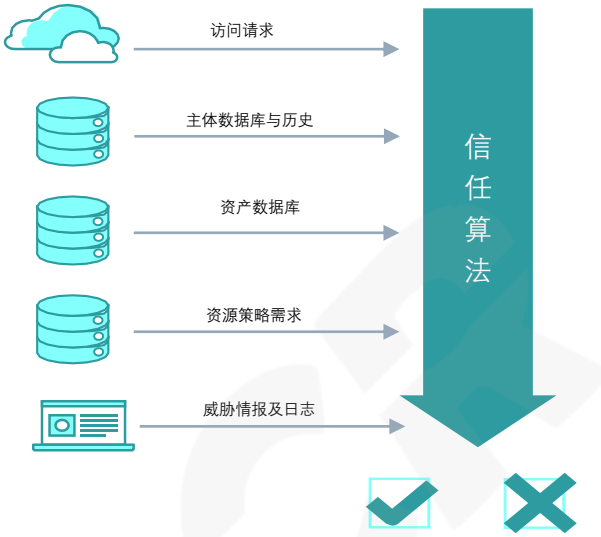


图9：来自 NIST 800-207 的信任算法图

NIST 800-207 文档中引入了基于信任算法评估的策略引擎的概念，如图 9 中的概念级别所示。

NIST 800-207 文档提到的策略模型中讨论了资源策略的要求：

> “这组策略补充了用户 ID 和属性数据库[SP800-63]，并定义了访问资源的最小要求。这些要求可能包括认证的基准级别，如 MFA 网络位置(例如，拒绝来自海外 IP 地址的访问)、数据敏感度和资产配置请求。这些要求应由数据保管人(即负责数据的人)和数据业务流程的负责人(即负责任务的人)共同制订。”

CISA 零信任成熟度模型并不直接对策略下定义，而是通过给出不同成熟度级别下的系统能力的具体示例来间接表明。

定义一个普遍适用的零信任策略模型的结构和词汇表是一件充满挑战的事情。这是“零信任”作为一个安全哲学思想所固有的本质。SDP 架构则更具体化，尽管不同 SDP 部署模型之间存在差异，但我们认为它非常适合结构化策略模型。。例如考虑一个设备安全态势检查的策略，该策略需要验证设备是否包含企业颁发的数字证书。对于 SDP 客户端软件运行在用户设备上的场景，证书验证很容易实施的[18]，但在无客户端的场景中就比较困难。这些实际场景中的限制导致行业研究和指南文件中(它们必须是与实现方案无关的)使用了更通用的策略表述（例如：确保用户设备满足企业安全态势检查)，而不是阐述更加明确和详细的策略模型。

零信任策略模型不在 SDP 标准规范 v2 的讨论范围内，但它是 SDP 和零信任工作组甚至整个行业未来的一个有趣研究领域。我们对于这个主题上未来的合作和成果发布充满期待。

使用单独分发的密钥）。

16 参见 NIST 特别出版物 800-207，零信任架构，第 6 页

17 美国网络安全和基础设施安全局网络安全部，零信任成熟度模型，决策前草案，2021 年 6 月，1.0 版本，第 5 页。

18 且在某些操作系统中，如果没有代理则可能无法实现。

3. SDP 协议

这里定义的 SDP 协议包括四个部分:AH–控制器协议、IH–控制器协议-、IH–AH 协议和日志，下面将详细说明。下文描述的 SDP 协议都展示了在 SDP 组件之间使用 SPA 进行通信，但它不是统一的，因为不同的 SDP 部署模型将需要不同的交互和消息。这里的目标是展示一个可运行的典型系统，而不是指定固定或标准化的消息格式。

值得注意的是，如前所述，该协议不包括组件或设备加载过程的消息或数据流。

为了简化起见，这里描述的协议没有指明超时、重试或其他错误处理机制。另外，在 IH 登录过程中，控制器可能会进行额外的外部调用，例如调用企业身份管理系统进行 IH 身份认证。最后，请注意下文的示例描述了组件之间 TCP 和 TLS 连接的使用。如前所述，使用基于 UDP 的无连接协议(即 DTLS)也是可以接受的，但为了简化起见，下文就不进行赘述了。

3.1 接受主机 AH-控制器协议

3.1.1 接受主机 AH 到控制器的时序图

接受主机连接控制器的协议时序图如图 10 所示。

基于 UDP 的 SPA 报文是第一个报文，确保 SDP 控制器免受未经授权的访问。

图 10：接受主机 AH 连接到控制器

以下小节定义了 AH 和控制器之间传递的各种消息和格式。基本的消息格式如下：

指令编号 (8-bit)	指令数据 (指令长度)

a.单包授权（SPA）

SPA 报文由 AH 发送到控制器以请求连接，遵循本文前面讨论的格式。

b.打开连接并建立双向认证的通信

AH 发送 SPA 报文后，将尝试打开与控制器的 TCP 连接。 如果控制器确定 SPA 报文合法有效，它将允许建立此 TCP 连接。

接下来是建立 mTLS 连接所需的双向身份认证。

基于 UDP 的 DTLS 是一个逻辑连接，因为 UDP 是一个无连接协议。

c.登录（加入 SDP）请求消息

登录请求消息由 AH 发送到控制器以表明它正在运行状态中，并请求作为活跃的 AH 加入 SDP。请注意，AH 登录请求消息中有可能包括 AH 自身的标识和认证凭据，以便控制器识别认证。

0x00	无指令特定数据

d.登录（加入 SDP）响应消息

登录响应消息由控制器发送以表明登录请求是否成功。如果成功，则提供 AH 会话 ID。请注意，控制器可能会拒绝 AH 的登录请求，这可能是由于 1）无效的认证凭据；2）或者控制器可能受到系统授权许可或扩展规模的限制。在这两种情况下控制器会终止该 AH 的连接。

在这两种情况下，连接都将被拒绝，并且控制器将终止来自 AH 的 TCP 连接。

0x01	状态码（16 位）	AH 会话 ID（256 位）以及更新 AH 的 SPA 密钥和 TLS 密钥（可选项）

状态码取决于实现方案。AH 会话 ID 在系统日志中使用。JSON 规范及示例如下：

格式	示例
{ “session_id” : <256 bits>, “credentials”: [“spa_encryption_key”:<64bit>, “spa_hmac_key”: <64 bit>, “tls_key”: <file contents>, “tls_cert” <file contents>] }	{ “session_id”: “0x1234…””credentials”: [“spa_encryption_key”:”aldskf…”, “spa_hmac_key”: “asldjf…”, “tls_key”: “tls_key”, “tls_cert”: “tls_cert”] }

e.注销请求消息

注销请求消息由 AH 发送至控制器，表示 AH 不再可用，且不再接收来自控制器的其他消息。控制器不响应该请求， TLS 和 TCP 的连接必须由 AH 或控制器终止。

0x02	无指令特定数据

f.心跳消息

Keep-Alive 心跳消息由 AH 或控制器发出，表示其仍处于活跃状态。

0x03	无指令特定数据

g.AH 服务消息

服务消息由控制器发送，该消息用于通告 AH 其所保护的服务集。注意，该示例中服务被指向一个固定 IP 地址（往往 AH 是一个 SDP 网关的时候会出现这种情况）。服务也可能被指向主机名， AH 则需要解析这个主机名。如果 AH 是服务所在主机的一部分 (见图 1A)，则使用 id 或 name 字段标识 AH 所保护的服务，而不是使用 IP 地址。

0x04	JSON 格式定义的服务数组

JSON 规范如下：

格式	示例
{“services”: [“port”: <Server port>, “id”: <256-bit Service ID>, “address”: <Server IP>, “name”: <servicename> “type” : <protocol type tag >] }	{“services”: [“port”: “443”, “id”: “123445678”, “address”: “10.2.1.123”, “name”:“Marketing Web App”, “type”:“HTTPS”] }

注意，此规范既可以描述基于 TCP 或 UDP 的服务，也可以用于其他协议(如 ICMP)的服务。

h.自定义用途的保留消息

该指令（0xff）为保留指令，用于 AH 和控制器之间的非标准消息。

0xff	用户自定义

3.2 IH-控制器协议

IH-控制器协议利用网络路由和报文传递，其实现细节依赖于传输协议类型（如 TCP 有保障的传输，或 UDP 的发后不管模式）。

3.2.1 IH 到控制器时序图

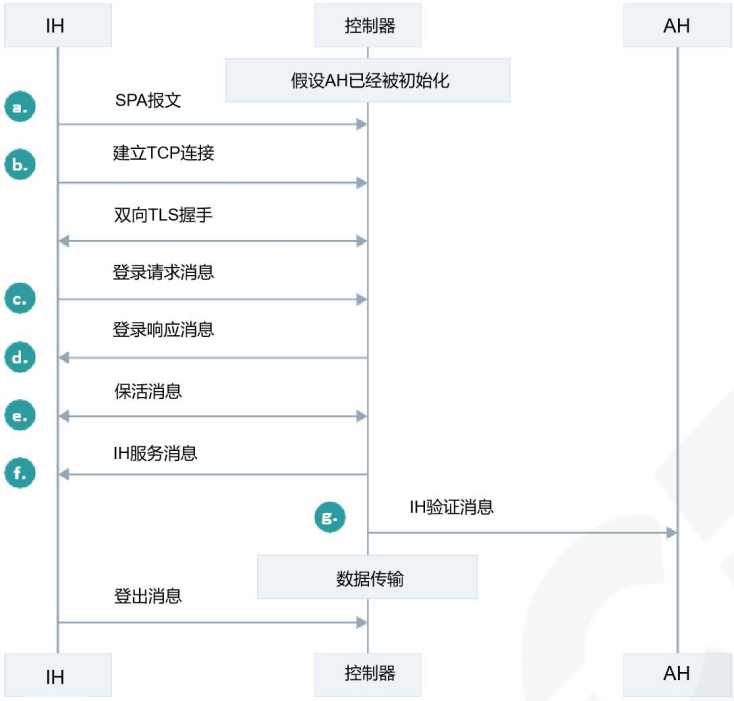


图 11：发起主机（IH）连接至 SDP 控制器

本节定义了 IH 和控制器之间交互的各种消息及其格式，基本协议格式如下：

指令(8 位)	特定长度的指令特定数据

a.SPA

IH 发送 SPA 报文到控制器请求连接，遵循本文前面讨论的格式.

b.打开连接并建立双向认证通信

IH 发送 SPA 报文后，将尝试打开与控制器的 TCP 连接. 如果控制器确定 SPA 报文合法有效，它将允许该 TCP 连接建立，随后进行建立 mTLS 连接所需的双向身份认证。

对于基于 UDP 的 DTLS 的情况，由于 UDP 是无连接协议，建立的是逻辑的连接。

c. 登录(加入 SDP)请求消息

IH 向控制器发送登录请求消息，以示 IH 可用并尝试加入 SDP。 注意, IH 向控制器发送的登录请求可能包含 IH 自身的身份标识和认证凭证。如前文所述，该登录请求出现在加载流程之后。该登录请求每次会话均会出现一次，例如在用户每天第一次打开他们设备的时候。

0x00	无特定指令数据

d.登录响应消息

控制器发送登录响应消息以指示登录请求是否成功。 如果成功，则向 IH 返回会话 ID。注意控制器可能因为某些原因拒绝 IH 登录请求，例如无效的认证凭据，或者控制器受到系统授权许可或扩展规模的限制。

0x01	状态码 (16 位)	IH 会话 ID (32 位)和 IH 的更新的 SPA 密钥和 TLS 密钥（可

		选项）

格式	示例
{ “session_id” : <256 bits>, “credentials”: [“spa_encryption_key”:<64 bit>, “spa_hmac_key”: <64 bit>, “tls_key”: <file contents>,“tls_cert” <file contents>] }	{ “session_id”: “0x1234…”“credentials”: [“spa_encryption_key”:”aldskf…”, “spa_hmac_key”: “asldjf…”, “tls_key”: “tls_key”, “tls_cert”: “tls_cert”] }

e.心跳消息

Keep-Alive 心跳消息由 IH 或控制器发送以表示仍处于活动状态。

0x03	无特定指令数据

f. IH 服务消息

服务消息由控制器发送，以向 IH 提供可用服务的列表以及保护这些服务的 AH 的 IP 地址或主机名. 此消息必须包含充分的信息，以便 IH 能够连接到该服务。请注意，列出的主机名/IP 地址是 IH 可直接访问的 AH。如前面的“部署模型”章节所述，实际服务可能运行在与 AH 不同的主机/IP 上。

服务 ID 将用于后续 IH 与 AH 通信时识别目标服务。

0x06	JSON 格式的服务数组

JSON 规范为：

格式	示例
{“services”: [{“address” : <AH IP>, “id”: <256-bit Service ID>, “name”: <service name>, “type” : <service type>, “port”: <Server port> }] }	{“services”: [{“address” : “10.2.1.34”, “id”: “123445678”, “name”: “FinanceApp”, “type” : “HTTPS”, “port” : “8443” }] }

g.IH 认证信息

接受主机的作用是确保受保护资源在被允许访问之前成功进行身份验证。控制器向 AH 发送 IH 的认证消息，以指示 AH 新的 IH 已被成功验证，且指示 AH 应允许 IH 访问指定的服务。

请注意，尽管此消息是从控制器发送到 AH 的，但它是通过 IH 向控制器进行身份验证而发起的。

0x05	IH 的 JSON 格式的服务数组信息

JSON 规范为：

格式	示例
{“IH Authenticators”: “IH”: <IH/Device Pair>, “session_id”: <256-bit IH Session ID>, “hmac_seed”: <256-bit SPA HMAC seed for the IH>, “hotp_seed”: <256-bit SPA HOTP seed for the IH> “id”: [<array of 256-bit service IDs>] }	{“IH Authenticators”: “IH”: “IH/DeviceID”, “session_id”: “4562”, “hmac_seed”: “###”, “hotp_seed”: “###”, “id”: [“123445678”, “9012345” }

请注意，控制器发送到 AH 的信息应该做够充分，以便让 AH 未来可以验证来自 IH 的连接。在上面的示例假设 AH 需要的 IH 专属的设备 ID、会话 ID 和单包授权密钥。不同的实现方案可能会以不同的方式达成这一目标。

h. 注销请求消息

注销请求消息由 IH 发送给控制器，以表明 IH 希望终止其 SDP 会话。控制器不用发送响应消息，TLS 和 TCP 连接必须由 IH 或控制器终止。请注意，IH 仍然是在加载状态，并且可以在将来再次建立新的会话。

0x07	无特定指令数据

z.自定义用途的保留消息

此指令（0xff）为保留指令，用于 IH 和控制器之间的任何非标准消息。图 11 中没有显示此消息。

0xff	用户自定义

3.3 IH–AH 协议

IH 到 AH 协议利用网络路由和报文传递。实现细节取决于传输的类型（例如，TCP 的有保障传输，或 UDP 的发后不管模式）。

需要注意的是，只有在 IH 使用有效的 SPA 单包授权报文连接、建立了双向认证的通信，并

且 AH 验证 IH 有权访问所请求的服务之后，才能通过 AH 动态地建立与服务的连接。否则，该服务一直被 AH 隐藏不可见。

3.3.1 IH 到 AH 序列图

图 12 显示了 IH 和 AH 之间协议的示例序列图。

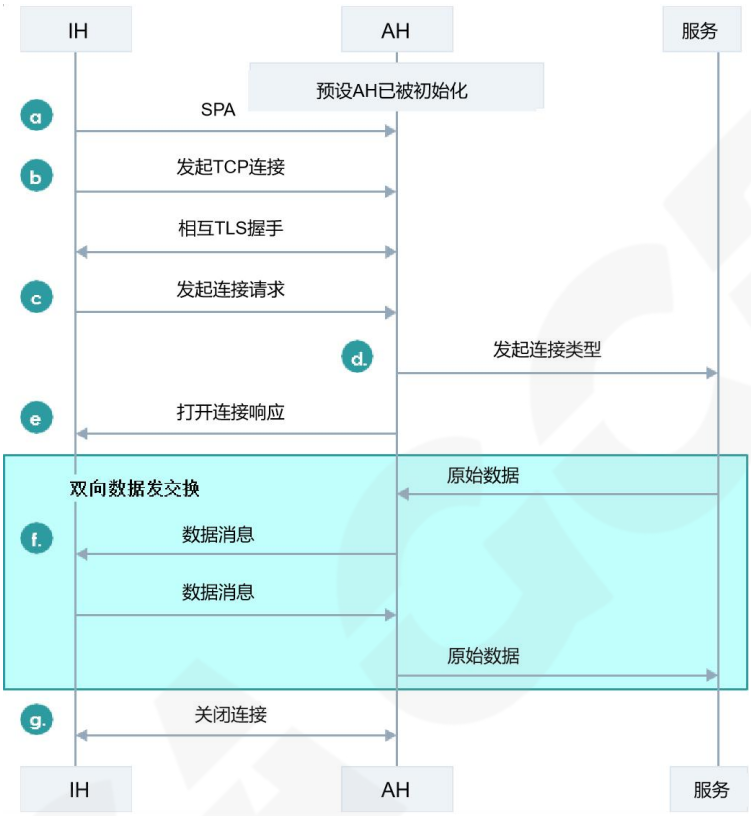


图 12：IH 连接到一个 AH，接着把数据送到一个服务

本节定义不同的消息和格式在 IH 和 AH 间交换。基本的协议如以下的格式：

指令（8 位）	根据指令类型有不同长度和数据内容

a.SPA

IH 发送 SPA 包到 AH，按照前面讨论的格式请求连接。

b.打开连接和建立双向认证的通信

IH 在发送 SPA 包后，会尝试打开与 AH 的 TCP 连接。如果 AH 判断 SPA 是合法有效的，AH 会允许 TCP 连接建立。随后进行建立 mTLS 连接所需的双向身份认证。

对于基于 UDP 的 DTLS 的情况，由于 UDP 是无连接协议，建立的是逻辑的连接

c. 发起连接请求的消息

当 IH 要连接到一个特定的服务，IH 会发送一个连接请求消息到 AH。

服务标识是一个独一无二的数值，由控制器生成。而 IH 能识别这些服务 ID 是因为这些标识会被控制器放入 IH 服务消息，并在更早之前就发送给 IH。（IH-控制器协议说明在前面有提及。）

会话标识被 IH 和 AH 用于区分不同远程服务的 TCP 连接。

0x07	服务标识 – 256 位 会话标识 – 256 位

d. 打开适合连接类型

这个步骤中，AH 会代表 IH 建立一个连接到服务。但该步骤会因为具体 SDP 实现方案和部署模型而有所不同。对某些协议来说，这个步骤可能是不必要的，比如短连接的 HTTPS，或是需要 IH 提供应用层认证来作为初始交互的那一类型连接（如 SSH）。SDP 感知服务可能需要借助这个连接从 AH 那里获得应用协议之外的 IH 或用户上下文信息。同时，取决于 SDP 的部署模型，这个连接有可能是一个网络上的连接或是在本机上的连接（如进程间通信）。

e. 打开连接响应消息

AH 会发送一个“打开–响应“消息给 IH 来告知连接请求是否成功。打开连接请求和打开连接响应消息对 IH 而言可能是异步的，因此服务标识和会话标识可以帮助 IH 把返回值和其对应的请求给关联起来。

0 x0 8	状态码（16bits）	服务标识 - 256 位 会话标识 - 256 位

f. 数据消息

数据消息由 IH 或 AH 发送，它用于在连接建立后推送数据。该消息无需回复。这个消息和具体 SDP 实现方案有关，有些 SDP 实现方案可能使用以下的格式，但也可以是其他替代的方案。

0 x 09	数据长度（16 位）	服务标识 - 256-位 会话标识 - 256-位

g. 连接关闭消息

连接关闭消息由 IH 或 AH 发送。它用于指示 AH 已关闭指定的连接，或 IH 正在请求关闭指定连接。该消息无需回复。

0x0A	服务标识 - 256 位标识符 服务标识 - 256 位标识符

z. 用户自定义消息

该指令（0xFF）为保留指令，用于表示 IH 和 AH 之间的非标准消息。图 12 中未显示此消息。

0xFF	用户自行定义

3.4 日志

日志的目的是为了确定服务的可用性和性能，以及服务器的安全性，它是所有系统和零信任实现方案的必须要求。

3.4.1 日志消息字段

所有日志均应包含以下基本字段。

字段	含义
时间	当前日志记录的产生时间
名称	人工可读的事件名称。注意：请勿包含任何变量数据，如用户名、IP 地址、主机名等，这些信息已包含在日志记录的内容信息字段中。
严重等级	事件的严重等级（从 debug 调试级到 critical 严重级，参看下文）
设备地址	产生该日志的设备 IP 地址

3.4.2 运维日志

下表给出了需要被记录的运维用例和活动清单。

特征标识（signature_id）是一个标识符，使之可以识别事件的类型。第三列包含需要记录的具体消息的额外字段。

活动	特征标识	记录的数据/信息
组件启动、关闭、重启（例如，控制器启动、主机重启）	ops:startup ops:shutdown ops:restart	**原因：**组件重启或关闭的原因； **组件：**指明受影响的组件。
组件（控制器、IH、AH、第三方组件、数据库）之间的连接事件，包括发起，断开和重连	ops:conn:up ops:conn:down ops:conn:reconnect	**源地址：**通信连接的源 IP 地址（以日志报告方为观察视角）； **目的地址：**通信连接的目标 IP 地址（以日志报告方为观察视角）； **重连次数：**尝试重新连接的次数； **原因：**说明通讯中断的原因。

3.4.3 安全/连接日志

安全日志是 SDP 的核心，在更广泛的场景下对于检测大规模的针对基础设施的攻击也很重要。因此将安全日志转发给安全信息和事件管理（SIEM）系统是非常有用的。它们是零信任实现方案的必要条件。

活动	特征标识	记录的数据/信息
AH 登录成功	*sec:login*	**源地址：**控制器所看到的 AH 的 IP 地址。 **AH Session ID：**AH 会话 ID。
AH 登录失败	*sec:login_failure*	**源地址：**控制器所看到的 AH 的 IP 地址。 **AH Session ID：**AH 会话 ID。

IH 登录成功	*sec:login*	**源地址：**控制器所看到的 IH 的 IP 地址。 **IH Session ID：**IH 会话 ID。
IH 登录失败	*sec:login_failure*	**源地址：**控制器所看到的 IH 的 IP 地址。 IH Session ID：IH 会话 ID。
SPA 认证	*Sec:auth*	从 IH 到控制器的 **SPA 报文** 从 AH 到控制器的 **SPA 报文** 从 IH 到 AH 的 **SPA 报文**
组件认证（如 IH ->控制器）	*Sec:connection*	**IH Session ID：** IH 的会话 ID **AH Session ID：**AH 的会话 ID
被拒绝的入站连接	*sec:fw:denied*	**源地址：**尝试连接的源 IP 地址。 **目标地址：** 尝试连接的目标地址

这里有个简单的场景描述了一个完整的故障，展示了写入了哪些日志条目以及写入位置。在这个示例场景下，我们假设控制器发生故障。

控制器出现故障 [无日志，故障组件无法记录]

1. IH 尝试多次重新连接控制器

   ops:conn:reconnect 日志消息
2. 经过多次尝试后，客户端声明失去与控制器的连接，并寻找新的控制器

   ops:conn:down 日志消息，严重性等级为错误（error）
3. IH连接到新发现的控制器

   ops:conn:up 日志消息
4. 如果没有更多控制器可用

   ops:conn:down 日志消息，严重性等级为极其严重（critical）

另一个类似的场景是客户端在没有警告的情况下关闭（例如，笔记本电脑被关闭）。在这种场景下，控制器和 AH 会检测到失败的连接。每个都会记录一条 ops:conn:down 日志消息，并严重等级标记为错误（error）。

下面是一个完整的用户登录（IH 连接到 AH）的示例：

5. IH连接控制器

   sec:auth 日志消息-对控制器的 SPA 身份验证

   ops:conn:up 日志消息
6. IH向Controller双向认证

   sec:连接 日志消息
7. IH 连接到 AH

   sec:auth 日志消息

   ops:conn:up 日志消息

总结

SDP 是一种有效的零信任实现方案[19]。修订后的标准规范将促进企业采用零信任范式来保护其应用程序、网络、用户和数据的安全。这一点变得至关重要，因为企业正在向云计算的迁移以及威胁形势正在不断加剧。

此外，SDP 已被证明可以保障企业的 IaaS 平台、网络功能虚拟化（NFV）、软件定义网络（SDN）和物联网 IoT 应用程序的安全。

软件定义边界 (SDP) 标准规范的 2.0 版已经酝酿了很久。1.0 版大约发布于 8 年前，即 2014 年 4 月。在此期间，我们看到了行业对零信任理念的热情拥抱以及越来越高的采用率。同时，市场上对于 SDP 的解决方案的兴趣和部署实施也相应增长。

2.0 版标准规范对以下方面进行了扩展和修改（通过细化澄清和延展）：

- SDP 概念及其与零信任的关系
- SDP 架构和组件
- 加载和访问流程
- SPA 消息格式、UDP 的使用、及其替代方案
- 关于物联网设备和访问策略的初步讨论

此外，我们在以下三个 SDP 子协议中提供了增强的序列图以及连接和消息说明：

- AH 到控制器
- IH 到控制器
- IH 到 AH

以下是 SDP 工作组正在考虑的未来研究和成果发布的主题，包括：

- 零信任策略模型和 SDP（即，通过策略决策点（PDP）和相应的策略执行点（PEP）进行访问）
- 使用 SDP 实现物联网零信任安全
- SDP 密钥的安全存储和轮换
- 设备校验

我们对于未来关于上述主题的合作和成果发布充满期待。

19 https://www.helpnetsecurity.com/2020/05/29/sdp-zero-trust/

参考文献

Cloud Security Alliance (CSA), Integrating SDP and DNS: Enhanced Zero Trust policy enforcement - Pending publication Q1 2022 available at https://cloudsecurityalliance.org/artifacts/integrating-sdp-and-dns-enhanced-zero-trust-policy-enforcement/

Cloud Security Alliance (CSA), Software Defined Perimeter (SDP) and Zero Trust, May 27, 2020, available @ https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-and-zero-trust/

Cloud Security Alliance (CSA), Zero Trust Presentation to OMG (Object Management Group), SDP: The Most Advanced Zero Trust Architecture, available @ https://cloudsecurityalliance.org/artifacts/sdp-the-most-advanced-zero-trust-architecture/

Cloud Security Alliance (CSA), SDP Architecture Guide version 2.0, published May 2019, available @ https://cloudsecurityalliance.org/artifacts/sdp-architecture-guide-v2/

Cloud Security Alliance (CSA), SDP Glossary, published June 2018, available @ https://downloads.cloudsecurityalliance.org/assets/research/sdp/SDP-glossary.pdf

Cloud Security Alliance (CSA), SDP as a DDoS Defense Mechanism, published October 2019, available @ https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-as-a-ddos-prevention-mechanism/

Waverley Labs, SDP Center, Open Source Reference Implementation (funded by DHS), available @ http://sdpcenter.com/test-sdp/

Cloud Security Alliance (CSA) Software-Defined Perimeter (SDP) Specification 1.0, published April 2014, available@ https://cloudsecurityalliance.org/artifacts/sdp-specification-v1-0/

Zero Trust Security: An Enterprise Guide, by Jason Garbis and Jerry W. Chapman, Apress, 2021, available @ https://www.apress.com/us/book/9781484267011

Zero Trust Networks: Building Secure Systems in Untrusted Networks, by Evan Gilman and Doug Barth, O'Reilly, 2017, available @ https://www.oreilly.com/library/view/zero-trust-networks/9781491962183/

National Institute of Standards and Technology (NIST), NIST SP 800-207, Zero Trust Architecture document, August 2020, available @ https://csrc.nist.gov/publications/detail/sp/800-207/final

Cybersecurity and Infrastructure Security Agency Cybersecurity (CISA) Division, Zero Trust Maturity Model (Draft), June 2021 Version 1.0 (Comment Period Ends October 1, 2021), available @ https://www.cisa.gov/publication/zero-trust-maturity-model

Office of Management and Budget, Federal Zero Trust Strategy (Draft), (Comment Period Ends

September 21, 2021, available @ https://zerotrust.cyber.gov/federal-zero-trust-strategy/

Cybersecurity and Infrastructure Security Agency (CISA), in collaboration with the United States Digital Service and FedRAMP, (Comment Period Ends October 1, 2021), available @ https://zerotrust.cyber.gov/cloud-security-technical-reference-architecture/

Executive Order 14028 on Improving the Nation's Cybersecurity, May 12, 2021, available @ https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/

National Security Agency (NSA), Embracing a Zero Trust Security Model, February 2021, available @ https://www.nsa.gov/Press-Room/Cybersecurity-Advisories-Guidance/smdpage11747/2/

Adoption of the Software-Defined Perimeter (SDP) Architecture for Infrastructure as a Service, by J. Singh, A. Refaey and J. Koilpillai, in Canadian Journal of Electrical and Computer Engineering, vol. 43, no. 4, pp. 357-363, Fall 2020, doi: 10.1109/CJECE.2020.3005316. https://ieeexplore.ieee.org/abstract/document/9240082

On IoT applications: a proposed SDP framework for MQTT, Electronics Letters, by Refaey, A.; Sallam, A.; Shami, A.: 2019, 55, (22), p. 1201-1203, DOI: 10.1049/el.2019.2334IET Digital Library, https://digital-library.theiet.org/content/journals/10.1049/el.2019.2334

P. Kumar, Abdallah Moubayed, Ahmed Refaey, Abdallah Shami, and Juanita Koilpillai "Performance Analysis of SDP for Secure Internal Enterprises," IEEE Wireless Communications and Networking Conference (WCNC), 1-6, 2019.

J. Singh, A. Refaey and A. Shami, "Multilevel Security Framework for NFV Based on Software Defined Perimeter," in IEEE Network, vol. 34, no. 5, pp. 114-119, September/October 2020, doi: 10.1109/MNET.011.1900563.

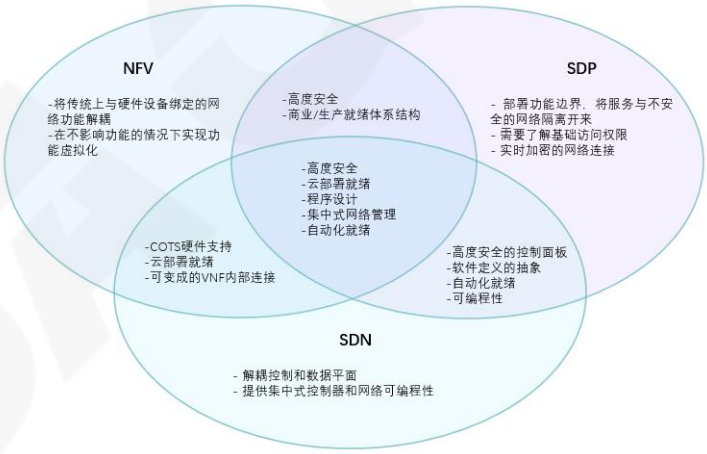
Ahmed Sallam, Ahmed Refaey, and Abdallah Shami, " On the Security of SDN: A Completed Secure and Scalable Framework Using the Software-Defined Perimeter," IEEE Access, Accepted, August 2019. (Impact factor: 4.098).

附录 A：SDP，SDN 和 NFV

在云计算环境中，软件定义网络（SDN）和网络功能虚拟化（NFV）技术有效解决了 IH 环境（前端）和 AH 环境（后端）的挑战[20]，同时提供了按需扩展、按需付费、并将资源作为服务提供的优势。SDN 和 NFV 为采用和编排异构网络路由以更好地利用资源（无线资源和计算资源）铺平了道路。IH 环境的挑战主要与保护无线移动网络或边缘网络的通信接入层有关，而 AH 环境的挑战主要在于路由、交换、安全、计费和收费以及其它诸如此类的网络路由功能所需的操作有关。

NFV 的主要挑战之一就是资源枯竭。密集使用特定物理服务器资源的软件可能会耗尽这些资源并且影响到虚拟机的可用性。出现这种情况是由于物理服务器中的共享环境加速了资源争用的严重性，特别是当多个虚拟机同时运行相同的资源消耗型软件的时候。这个问题可以通过采用 SDP 得到解决，通过让 SDP 控制器定义并实施一个标准操作程序，以检测那些资源耗尽而受到节流的虚拟机（类似于 DoS），并动态采取补救措施。NVF 中的另一个常见风险是账户或者服务通过自助服务门户或云管理控制台被黑客劫持，因为授予终端用户过多的管理权限都会增加门户站点或控制台暴露的风险。对于这种情况，SDP 基于用户的角色及职能选择性地赋予其管理控制权限，因此在消除此类风险方面发挥了至关重要的作用。

云服务提供商广泛采用软件定义网络（SDN）来简化网络管理。SDNs 的主要挑战是如何为 API 驱动的网络路由编排提供适当的身份验证、访问控制、数据隐私以及数据完整性等。在此，软件定义边界（SDP）能够提供连接的编排，限制网络访问以及支持 SDN 网络基础设施对象之间的安全连接。



SDP 和 SDN 的集成有许多潜在优势，尤其是它提供了一个完全可拓展和可管理的安全解决方案。

简而言之，可以将软件定义网络（SDN）和网络功能虚拟化（NFV）视为网络虚拟化三角形的两个边，而软件定义边界（SDP）则是第三条边。尽管 SDN 和 SDP 都在网络层中运行且名称近似，但 SDN 可以被视作协调网络运作的大脑，而 SDP 则以零信任的概念引入可靠的网络连接，两者融合没有明显障碍。

20 J. Singh、A. Refaey 和 J. Koilpillai，“为基础设施即服务采用软件定义边界 (SDP) 架构”， 在 2020 年秋季的《加拿大电气与计算机工程杂志》43 卷，4 号，第 357-363 页，doi：10.1109/CJECE.2020.3005316。

附录 B：OSI/SDP 组件映射

本节提供 OSI 网络层到云计算层、云堆栈功能和 SDP 组件的映射。该映射用于展示 SDP 是如何为不同的网络层和云模型层次提供安全性。

OSI 层	云计算层	云堆栈	SDP 组件
应用层	应用程序	终端用户层：提供应用程序和商业价值	SDP 客户端：提供对应用程序的用户访问
表示层	服务	中间件：应用程序在各层级中使用的功能性组件	SDP 控制器：充当用户访问资源的代理中间件
会话层	镜像	操作系统：合理管理底层虚拟化	SDP 策略：执行防火墙规则和会话管理
传输层	软件定义数据中心	云 API：支持创建与资源池/用户绑定的虚拟资产	双向 TLS
网络层	管理程序	虚拟化：提供计算、储存和监控的虚拟化	SDP 接受主机（网关）
数据链路层	基础设施	硬件：数据中心的物理设备	（不适用）
物理层	基础设施	硬件：数据中心的物理设备	（不适用）