Software-Defined Perimeter (SDP) 仕様書 v2.0

CSAJC cloud security alliance SM Japan Chapter

CSA cloud security alliance®

Software Defined Perimeter Working Group の恒久的かつ公式な場所は、
https://cloudsecurityalliance.org/research/working-groups/software-defined-perimeter-and-zero-trust/ です。

Acknowledgments

Lead Authors

Jason Garbis
Juanita Koilpillai

Contributors

Junaid Islam
Bob Flores
Daniel Bailey
Benfeng Chen
Eitan Bremler
Michael Roza
Ahmed Refaey Hussein

Special Thanks

Larry Hughes

Version 1.0 Contributors

Brent Bilger, Alan Boehme, Bob Flores, Zvi Guterman, Mark Hoover, Michaela Iorga, Junaid Islam, Marc Kolenko, Juanita Koilpillai, Gabor Lengyel, Gram Ludlow, Ted Schroeder and Jeff Schweitzer

Reviewers

Alistair Cockeram
Takahiro Ono
T Prasad
Nya Murray
Michael Rash

CSA Analyst

Shamun Mahmud

CSA Global Staff

Claire Lehnert

CSA Analyst

The Software-Defined Perimeter (SDP) and Zero Trust Working Group is a Cloud Security Alliance (CSA) research working group that advocates for and promotes the adoption of Zero Trust security principles. It provides practical and sound guidance on how to apply Zero Trust to cloud and non-cloud environments. This group will build on and leverage the NIST Zero Trust research and methodologies. It will recommend SDP as a fundamental architecture for applying Zero Trust
principles. It will revise and expand the SDP specification, to capture and codify a wealth of knowledge based on experience. Lastly, the SDP group recognizes the need for an inclusive approach to SDP. As such, it also supports alternative architectures so long as they align with Zero Trust principles.

Dedication

This paper is dedicated to the memory of Juanita Koilpillai, a security leader, influencer, mentor, and friend. Juanita was a luminary in her field and broke many glass ceilings on her path to becoming founder and chief executive at two successful start-ups. In return, Juanita shared her knowledge by mentoring young women with careers in technology. She played a prominent mentorship role in IEEE's Women In Engineering (IEEE WIE) Organization. She also created a philanthropic fund in memory of her father and founded MERGE, a non-profit to support local youth projects. She was instrumental in the authorship of this as well as several other CSA, IEEE and NIST publications.

日本語版提供に際しての告知及び注意事項

本書「Software-Defined Perimeter (SDP) 仕様書 v2.0」は、Cloud Security Alliance (CSA)が公開している「Software-Defined Perimeter (SDP) Specification v2.0」の日本語訳です。本書は、CSAジャパンが、CSAの許可を得て翻訳し、公開するものです。原文と日本語版の内容に相違があった場合には、原文が優先されます。
翻訳に際しては、原文の意味および意図するところを、極力正確に日本語で表すことを心がけていますが、翻訳の正確性および原文への忠実性について、CSAジャパンは何らの保証をするものではありません。
この翻訳版は予告なく変更される場合があります。以下の変更履歴(日付、バージョン、変更内容)をご確認ください。

変更履歴

日付	バージョン	変更内容
2022年05月04日	日本語版1.0	初版発行

CSAジャパン成果物の提供に際しての制限事項

日本クラウドセキュリティアライアンス(CSAジャパン)は、本書の提供に際し、以下のことをお断りし、またお願いします。以下の内容に同意いただけない場合、本書の閲覧および利用をお断りします。

1. 責任の限定

CSAジャパンおよび本書の執筆・作成・講義その他による提供に関わった主体は、本書に関して、以下のことに対する責任を負いません。また、以下のことに起因するいかなる直接・間接の損害に対しても、一切の対応、是正、支払、賠償の責めを負いません。

(1) 本書の内容の真正性、正確性、無誤謬性
(2) 本書の内容が第三者の権利に抵触しもしくは権利を侵害していないこと
(3) 本書の内容に基づいて行われた判断や行為がもたらす結果
(4) 本書で引用、参照、紹介された第三者の文献等の適切性、真正性、正確性、無誤謬性および他者権利の侵害の可能性

2. 二次譲渡の制限

本書は、利用者がもっぱら自らの用のために利用するものとし、第三者へのいかなる方法による提供も、行わないものとします。他者との共有が可能な場所に本書やそのコピーを置くこと、利用者以外のものに送付・送信・提供を行うことは禁止されます。また本書を、営利・非営利を問わず、事業活動の材料または資料として、そのまま直接利用することはお断りします。

ただし、以下の場合は本項の例外とします。

(1) 本書の一部を、著作物の利用における「引用」の形で引用すること。この場合、出典を明記してください。

(2) 本書を、企業、団体その他の組織が利用する場合は、その利用に必要な範囲内で、自組織内に限定して利用すること。
(3) CSA ジャパンの書面による許可を得て、事業活動に使用すること。この許可は、文書単位で得るものとします。
(4) 転載、再掲、複製の作成と配布等について、CSA ジャパンの書面による許可・承認を得た場合。この許可・承認は、原則として文書単位で得るものとします。

3. 本書の適切な管理

(1) 本書を入手した者は、それを適切に管理し、第三者による不正アクセス、不正利用から保護するために必要かつ適切な措置を講じるものとします。
(2) 本書を入手し利用する企業、団体その他の組織は、本書の管理責任者を定め、この確認事項を順守させるものとします。また、当該責任者は、本書の電子ファイルを適切に管理し、その複製の散逸を防ぎ、指定された利用条件を遵守する（組織内の利用者に順守させることを含む）ようにしなければなりません。
(3) 本書をダウンロードした者は、CSA ジャパンからの文書（電子メールを含む）による要求があった場合には、そのダウンロードしまたは複製した本書のファイルのすべてを消去し、削除し、再生や復元ができない状態にするものとします。この要求は理由によりまたは理由なく行われることがあり、この要求を受けた者は、それを拒否できないものとします。
(4) 本書を印刷した者は、CSA ジャパンからの文書（電子メールを含む）による要求があった場合には、その印刷物のすべてについて、シュレッダーその他の方法により、再利用不可能な形で処分するものとします。

4. 原典がある場合の制限事項等

本書がCloud Security Alliance, Inc.の著作物等の翻訳である場合には、原典に明記された制限事項、免責事項は、英語その他の言語で表記されている場合も含め、すべてここに記載の制限事項に優先して適用されます。

5. その他

その他、本書の利用等について本書の他の場所に記載された条件、制限事項および免責事項は、すべてここに記載の制限事項と並行して順守されるべきものとします。本書およびこの制限事項に記載のないことで、本書の利用に関して疑義が生じた場合は、CSAジャパンと利用者は誠意をもって話し合いの上、解決を図るものとします。

その他本件に関するお問合せは、info@cloudsecurityalliance.jp までお願いします。

日本語版作成に際しての謝辞

「Software-Defined Perimeter（SDP）仕様書 v2.0」は、CSAジャパン会員の有志により行われました。作業は全て、個人の無償の貢献としての私的労力提供により行われました。なお、企業会員からの参加者の貢献には、会員企業としての貢献も与っていることを付記いたします。
以下に、翻訳に参加された方々の氏名を記します。（氏名あいうえお順・敬称略）

小野　貴博
小田部　悟士
髙岡 隆佳
諸角　昌宏

目次

はじめに

Software-Defined Perimeter (SDP) アーキテクチャは、ネットワークセキュリティの境界を動的かつ柔軟に展開し、安全が担保されていないネットワークにあるアプリケーションやサービスを隔離する機能を提供します。SDP は、隔離に必要な、オンデマンドで、かつ動的にプロビジョニングされる信頼のオーバーレイを提供し、企業が管理するネットワークの内外からのネットワークベースの攻撃を軽減するように設計されています。SDPの実装では、不正な存在から資産を隠し、接続を許可する前に信頼を確立し、分離されたコントロールプレーンとデータプレーンによってシステムを管理します。SDPの導入によって、企業はゼロトラストの目標を達成することができ、その結果、従来の（効果がなくなりつつある）境界中心のモデルから脱却し、セキュリティの有効性と回復力を向上させることができます。

目的

本仕様は、2014年4月にSoftware Defined Perimeter Working Group（SDP WG）が公開したCloud Security Alliance（CSA） Software-Defined Perimeter（SDP）Version 1.0 を更新したものです。

当初の仕様も安定したものでしたが、コンポーネントのオンボーディングや、ノンパーソンエンティティ[1]の保護など、いくつかのトピックに十分対応していませんでした。また、SDP v1のリリース後、業界はSDPの原則を積極的に受け入れ、現在私たちが*ゼロトラスト*と呼ぶものに取り込んできました。この改訂版では、より多くのトピックへの対応や記述内容の改善に加え、ゼロトラストをめぐる業界の現状を反映しています。

この改訂版は、SDP WGがSDP v1以降に公開した文書、特にSDP用語集 とSDPアーキテクチャガイドに基づいています。これら2つの文書へのリンクは、参考文献のセクションに記載しています。

スコープ

本仕様は、SDPにおける、アーキテクチャの構成要素、相互作用、基本的なセキュリティ通信プロトコルを扱います。セキュリティ境界内の安全な接続を可能にするコントロールプレーンと、サーバー、デバイス、サービスなどの*initiating host*と*accepting host*の間の安全な接続を実現するデータプレーンに焦点を当てます。

想定する読者

本仕様の対象読者は以下の通りです。

- ゼロトラストやSDP製品を企業で導入しているソリューションアーキテクトおよびセキュリティリーダー
- SDPアーキテクチャを使ってゼロトラストを製品に実装しているベンダーやテクノロジープロバイダ

1 **ノンパーソンエンティティ**（NPE）とは、**人間**ではないが、デジタル・アイデンティティを持って、サイバー空間で活動する**実体**のことです。ハードウェア装置、ソフトウェアアプリケーション、および情報成果物などが含まれます。

SDPのデザイン

SDPは、企業のセキュリティアーキテクト、ネットワークプロバイダ、アプリケーション所有者に、次のような機能を提供することを目的としています。

- 動的な「ソフトウェア定義」境界の展開
- ネットワークとリソースの隠蔽
- SDP上で動作するサービスに対する不正アクセスの防止
- アイデンティティを中心としたアクセスポリシーモデルの実施

SDPは、物理的なアプライアンスを、物理的な配置場所に関わらず、組織の管理下で動作する論理的なコンポーネントに置き換えることにより、論理的な境界を最小化します。SDPは、最小権限によるアクセス、侵害の想定、「信頼するが検証する」といったゼロトラストの原則を実施し、認証とアイデンティティの確認後に初めて、ポリシーベースのリソースアクセスを許可します。

SDPの設計目的は、IPv4およびIPv6ネットワークに対して、効果的で容易に統合できるセキュリティアーキテクチャを提供することです。設計目的には、コントロールプレーンの要素の隠蔽およびアクセス制御を含みます。SDPは、データプレーンを流れる通信に機密性と完全性を提供します。また、アイデンティティ（ユーザーまたはノンパーソンエンティティ）の認証を、検証されたデバイス上で行い、境界に暗号技術を使ってサインインすることにより、Need-to-knowの原則に従ったアクセスモデルに対応します。

SDPの設計は、複数のレイヤーでシームレスに統合されることにより、ユーザー、ユーザーの機器、ネットワーク、デバイスに対するセキュリティを統合します。SDPは、従来のハードウェアベース、Software-Defined Network（SDN）、クラウドベースに関わらず、あらゆるIPインフラで接続の保護に使えます。SDPの相互認証トンネルは、有効な暗号化されたオーバーレイであり、あらゆる種類のIPネットワーク上に配置できます。その結果、SDPは異なる環境間でセキュリティのレイヤーを正規化し、ネットワーク、セキュリティ、運用を簡素化します。

クラウドインフラの場合、SDPは、OSIの7層のうち下記の5層にセキュリティを統合します。

- **ネットワーク層：**仮想化[2]によって、コンピューティング、ストレージ、モニタリングが提供されます。
- **トランスポート層：**クラウドAPIが、仮想化された資産をリソースプールやユーザーに紐付けます。
- **セッション層：**下の層にある仮想化されたインフラを管理します。
- **プレゼンテーション層：**ミドルウェアがアプリケーション層とアプリケーションを管理します。
- **アプリケーション層：**ユーザーにビジネス価値を提供します。

SDPが統合されないOSIの層は、データリンク層と物理層です。これらの層を統合すると、TCP/IPモデルのネットワーク層に相当します。

SDPとネットワーク層のモデルについては、付録Bを参照してください。

SDNやNetwork Function Virtualization（NFV）を補完するものとして、SDPはSDNが構築するIPインフラ上の接続を保護することができます。

2　SDPとNetwork Function Virtualization（NFV）に関する論文 - https://www.waverleylabs.com/resources/publications/

この更新版の仕様では、これまでのフィードバックとSDPの実装から得た教訓に基づいて、初版の仕様で定義し、SDPアーキテクチャガイド[3]で詳細に説明したさまざまな配備モデルを題材として、アクセス制御の課題を明確にしています。

SDPは、アプリケーションやインフラに対する、ネットワーク攻撃やクロスドメイン攻撃の防止、検知、対応において非常に優れたアプローチを提供します。これは、ユーザーがインターネットのような信頼されないパブリックネットワーク経由でリソースにアクセスする場合においても、攻撃対象面を最小化し、最小権限の原則を適用することによって実現されます。

従来のサイバーセキュリティソリューションがネットワークやシステムの保護に重点を置いているのに対して、SDPはアイデンティティ中心の視点で資産を保護することに重点を置いています。境界中心のセキュリティから移行することにより、企業は、企業が管理するプライベートなリソースに対する、DDoS、認証情報の窃取、ランサムウェアなどの攻撃への耐性を高めることができます。

SDPのコンセプト

SDPは、組織の境界ではなく、組織の重要なリソースを保護することに重点を置いています。リスクベースで、ダイナミックな、アイデンティティ中心かつコンテキストを意識したアクセスポリシーを、ネットワークのすべてのレイヤーに定義して、適用します。SDPは、ビジネス、アプリケーション、ネットワークの関係者にとって意味のある言語でアクセスポリシーを定義し、適用するための基盤を提供します。多くの場合、組織内で初めて実現することです。

SDPは、アプリケーションやエンタープライズリソースの所有者に、境界を配備する能力を提供します。

- 侵害されているかもしれないネットワークに、安全にサービスを配備します。（すなわち、「侵害を想定」します）
- アイデンティティに対して、信頼されないネットワーク経由のサービスへのアクセスをきめ細かく提供します。ユースケースの1つとして、VPNの代替があります。

SDPは、境界ベースの（多くの場合、物理的な）アプライアンスを、アプリケーション所有者の管理下で動作する論理的なコンポーネントに置き換えます。SDPは、アクセスポリシーに従って、デバイスの認証とアイデンティティの検証を行った後に初めて、アプリケーションへのアクセスを提供します。

SDPの背後にある原理は、完全に新しいものではありません。国防総省や情報機関の中の複数の組織で、ネットワークアクセス前の認証と認可を基礎として、このようなネットワークアーキテクチャを実装してきました。一般的に、国防総省が定義するような、機密性の高い、「high-side」ネットワークで使用され、すべてのサーバーはリモートアクセスゲートウェイアプライアンスの背後に隠蔽され、ユーザーは認可されたサービスを確認しアクセスを許可される前に、アプライアンスに対して認証を行う必要があります。SDP は、機密ネットワークで使用された論理モデルを応用し、標準的なワークフローを組み込みました。セキュリティのリーダー達は、2004年のジェリコフォーラムを皮切りに、長年に渡ってこれらのコンセプトの多くを支持してきました。最近では、米国国立標準技術研究所（NIST）が定義した「ゼロトラスト」に、これらの原則が採用されています[4]。

3 https://cloudsecurityalliance.org/artifacts/sdp-architecture-guide-v2/
4 NIST Zero Trust Architecture - SP 800-207 https://csrc.nist.gov/publications/detail/sp/800-207/final

SDPは、上記のneed-to-know（最小権限）モデルの利点を維持しつつ、リモートアクセスゲートウェイアプライアンスを必要とするデメリットを排除しています。実際、SDPは「リモート」ユーザーだけでなく、「すべての」ユーザーにアクセス制御を適用するように設計されています。SDPは、保護されたサーバーやサービスへのネットワークアクセスを得る前に、エンドポイントの認証と認可を要求します。その後、アクセス元のシステムとアプリケーションインフラとの間に、暗号化された接続をリアルタイムに作成します。つまり、SDPは、ユーザー、デバイス、サービスなどのリソースを境界で安全に認証し、未認可のリソースにはサービスを隠蔽し続けます。

SDPのアーキテクチャと構成要素

最も単純なケースでは、SDPは、SDPホストとSDPコントローラといった、2つの論理コンポーネントで構成されます。SDPホストは、フルスタックのホストでも軽量サービスでも良く、接続の開始または受入を行います。これらのアクションは、「コントロールプレーン」の安全なチャネルを介して、SDPコントローラとのやり取りによって管理されます。データは独立した「データプレーン」の安全なチャネルを通って通信されます。コントロールプレーンがデータプレーンから分離されているため、柔軟でスケーラビリティの高いアーキテクチャのシステムを実現することができます。さらに、スケーラビリティや可用性のため、すべてのコンポーネントを冗長化することができます。SDPホスト（Initiating もしくは Accepting）は、アプライアンスもしくはサーバープロセスであるSDPコントローラと協調し、ユーザーが認証・認可され、デバイスが検証され、安全な通信が確立され、ネットワーク上のデータとコントロール用のトラフィックが分離されることを確実にします。

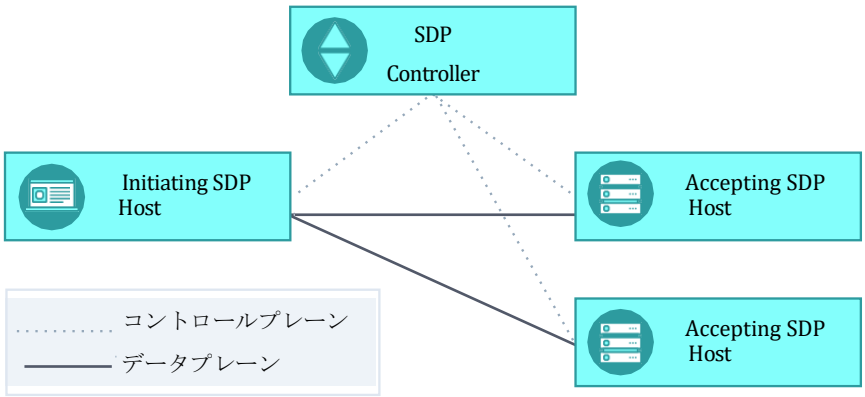


図1：SDPアーキテクチャ（CSAがSoftware Defined Perimeter and Zero Trustで既公開）[5]

SDPのアーキテクチャは、以下のコンポーネントから構成されます。

- **SDPコントローラ** ― このコンポーネントは、すべての認証とアクセスフローを管理するように設計されています。アクセスポリシーを定義し、評価するための、ソリューションの本質的な「頭脳」です。ゼロトラストのポリシー決定ポイント（PDP）として機能します。SDPコントローラは、企業の認証ソリューション（アイデンティティプロバイダ、多要素認証サービスなど）と連携し、認証と認可のオーケストレーションを担います。定義されたアクセスポリシーによって許可した接続を可視化し、レポートするためのコントロールポイントにもなります。
- **Initiating Host（IH）** ― これらのアクセス元のエンティティは、ユーザーのデバイスもしくは、ハードウェア（エンドユーザーデバイスやサーバーなど）、ネットワークデバイス、ソフトウェアアプリケーション、サービスなどのNPE（ノンパーソンエンティティ）です。SDPのユーザーは、SDPクライアントまたはブラウザを使用して、通信を開始することができます。

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

- **Accepting Host (AH)** ー これらのエンティティは、SDPによって保護およびアクセスされるアプリケーションや、サービス、リソースの前に存在する論理的なコンポーネントです。AHは、ゼロトラストのポリシー実施ポイント(PEP)として機能します。PEPは、SDP専用のソフトウェアもしくはハードウェアとして実装することができます。AHは、SDPコントローラからの指示に基づいて、対象サービス（アプリケーション、軽量なサービス、またはリソース）へのネットワークトラフィックを許可または拒否します。論理的に、AHは対象サービスと同居しても、 ネットワークで分離されていても構いません。

これらのSDPコンポーネントは、オンプレミスまたはクラウドに配置することができ、スケーラビリティや可用性のために冗長化して配置することができます。

ここから、各構成要素について見ていきます。

SDPコントローラ

SDPコントローラは、ポリシーを定義し、検証、決定するメカニズムであり、ゼロトラストのポリシー決定ポイントとして、どのアイデンティティ（ユーザーやグループなど）がどのデバイスから組織のサービス（オンプレミスまたはクラウド）にアクセスできるかという情報を管理します。そして、どのSDPホストが互いに通信できるかを決定します。

IH上のユーザーがコントローラに接続すると、コントローラは、ユーザーを認証し、アイデンティティやデバイス属性を含むユーザーのコンテキストに基づいて、ユーザーに許可されたサービスのみへのアクセスを（AHを介して）与えます。

ユーザーを認証するため、コントローラは内部のユーザーテーブルを使用したり、サードパーティーのIdentity and Access Management（IAM）サービス（オンプレミスまたはクラウド）に接続したり、多要素認証（MFA）を実施することができます。通常、認証はユーザーの種類やアイデンティティに基づいて処理されます。例えば、従業員はアイデンティティプロバイダを介して認証される一方、契約社員はデータベースに保存された認証情報やアイデンティティフェデレーションによって認証されるかもしれません。

ユーザーにサービスへのアクセスを認可するため、コントローラは内部のユーザー・サービスマッピングやポリシーモデル、もしくはLDAP、Active Directory、その他の認可ソリューションなどのサードパーティーのサービス（オンプレミスまたはクラウド）を使用することができます。通常、認可はユーザーの役割と、ユーザーやデバイスの属性、またはユーザーがアクセスする実際のデータ要素やデータフローに基づく詳細な情報によって決定されます。実際、SDPコントローラが保持するアクセスポリシーは、エンタープライズサービスディレクトリやアイデンティティストアなどの他の仕組みから情報を得ることがあります。このように、コントローラは、NISTがゼロトラストの理念に掲げる、動的なゼロトラストポリシーを実現します。

さらに、コントローラは、地理的な情報やホストの検証サービスなど外部サービスから情報を取得し、IH上のユーザーをさらに検証することができます。また、コントローラは、ユーザーの認証失敗や、機密性の高いサービスへのアクセスなど、他のネットワークコンポーネントにコンテキスト情報を提供することができます。

ゼロトラストのPDPに対応するSDPコントローラは、SDPのアーキテクチャに応じて、クラウドまたはオンプレミスに配置できます。

SDPコントローラは、Single Packet Authorization（SPA）プロトコルによる隔離メカニズムによって保護されており、未承認のユーザーやデバイスから隠蔽され、アクセスすることもできません。このメカニズムは、コントローラの前にSDPゲートウェイを設置して提供される場合もあれば、コントローラ自身が提供する場合もあります。

SDP Initiating Host (SDPクライアント)

SDPのIHはSDPコントローラと通信し、Accepting Hostを介して保護された企業リソースにアクセスするためのプロセスを開始します。

通常、コントローラは、認証フェーズにおいて、ユーザーのアイデンティティ、ハードウェアやソフトウェアのインベントリ、およびデバイスの健全性などの情報を提供することをIHに要求します。また、コントローラは、AHと安全な通信を確立するためのメカニズム（認証情報など）をIHに提供する必要があります。

IHは、エンドユーザーのマシンにインストールするクライアントアプリケーションの場合や、Webブラウザの場合があります。クライアントアプリケーションを使用することで、ホストチェック（デバイスポスチャーチェック）や、トラフィックルーティングなど、より合理的な認証を行うための豊富な機能が提供できます。

Initiating Host (IH)の最も重要な役割のひとつは、後述するように、SPAを使って接続を開始することです。実装によっては、ブラウザベースのSDPクライアントからSPAパケットを生成することもあります。IHは、人間が操作するユーザーデバイス（従業員や契約社員のコンピュータ、モバイルデバイスなど）、クライアントを組み込んだアプリケーション、IoTデバイス（リモート水道メーターなど）の場合があります。後半の例では、アイデンティティが人間ではありませんが、認証や認可が必要です。詳細は、「*SDPおよびIoT デバイス*」のセクションを参照してください。

SDP Accepting Host (AH)

Accepting Host (AH)は、必要なアイデンティティに接続を許す一方、企業が責任を持ってアクセスを隠蔽および制御するなど、SDPのポリシー実施ポイント（PEP）として、リソース（またはサービス）へのアクセスを制御します。AHは、オンプレミス、プライベートクラウド、パブリッククラウドなどに配置できます。

AHが保護するサービスは、Webアプリケーションに限らず、TCPやUDPベースのアプリケーション（SSH、RDP、SFTP、SMBなど）、クライアント・サーバーによる商用アプリケーションも考えられます。

AHへのすべてのネットワークアクセスはデフォルトでブロックされ、認証および許可されたエンティティのみがアクセスできます。

前述のように、AHは対象サービスと同居する場合もあれば、ネットワークを介して分離する場合もあります。これらのモデルを図2に示します。

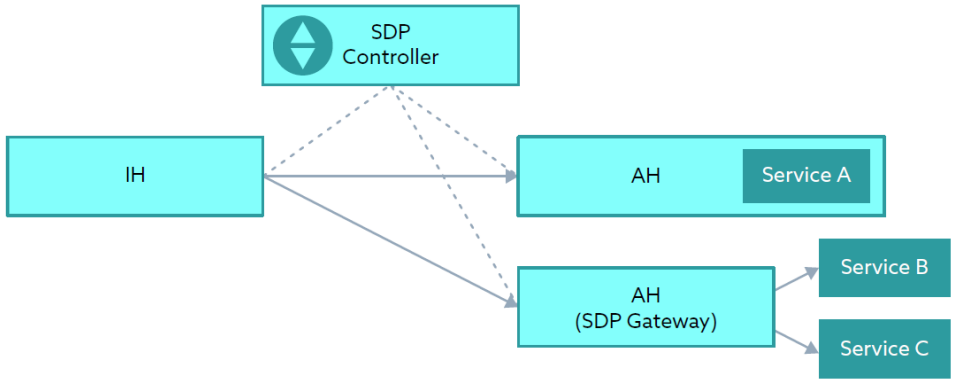


図2

AHはSDPコントローラから制御情報を受け取り、コントローラから指示された場合に限り、IHからの接続を受け入れます。AHは、SDPコントローラから受け取った制御情報を用いて、認可されたIH（ユーザやデバイス）に対して、保護されたサービスへのアクセスを提供します。

AHは、IHからトラフィックを受け取り、保護されたバックエンドサービスに中継するため、安全な通信やアクセスの交換機として機能します。バックエンドサービスからの応答は、IHに送り返されます。

SDPは論理的な接続を重視したプロトコルのため、さまざまなネットワーク配備のトポロジーに対応します。下図3に紹介する多様な配備モデルとアーキテクチャは、SDP アーキテクチャガイドに詳述されています。これらによって、さまざまな配備モデルに合わせたSDPコンポーネントの構成を示しています。

SDPの配備モデル

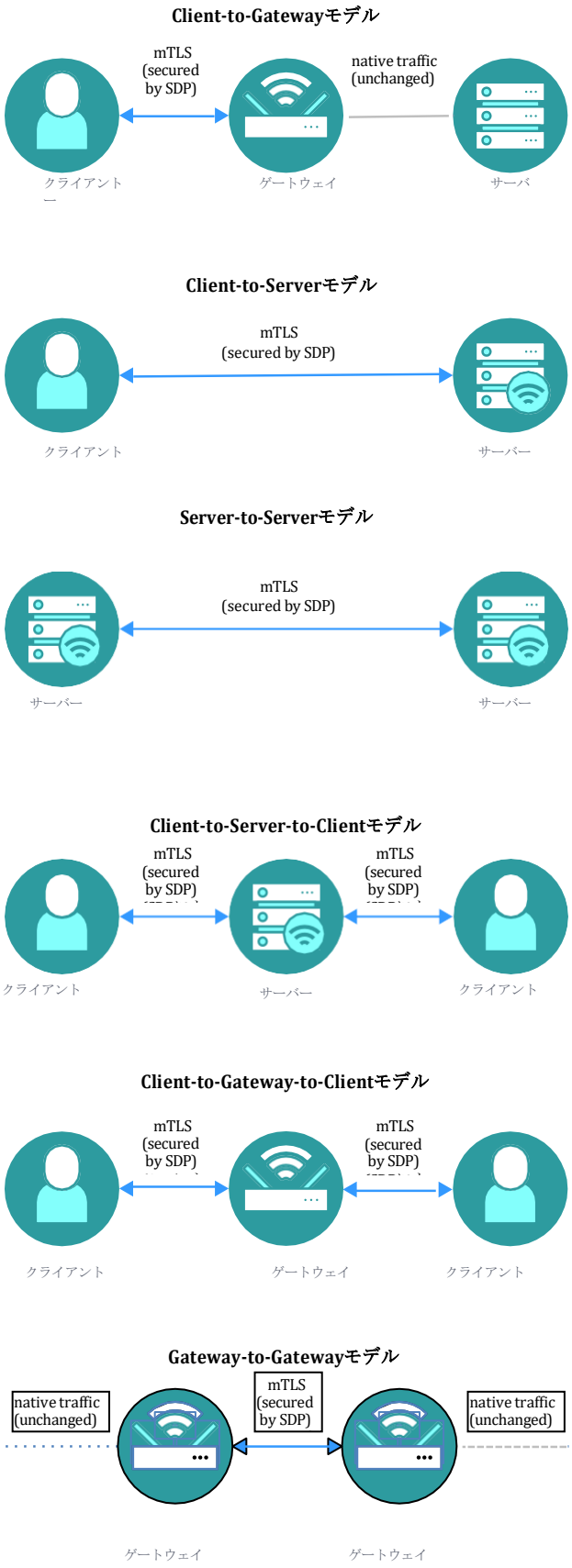
SDPは、クライアント（人間またはノンパーソンエンティティ）と、後述の図でサーバーとして表現されるリソースを接続します。これらのリソースは、ネットワークを介して通信することができれば、どのようなものでも構いません。物理サーバーや仮想サーバー上で動作するサービスであったり、IaaSやPaaSプラットフォーム上で動作するサービス、コンテナ化されたサービスなどかもしれません。

このセクションでは、簡単に6つのSDP配備モデルを紹介します。これらは異なるネットワークトポロジーですが、保護されたリソースへのアクセスを制御するという点で論理的に同じです。これらの配備モデルの詳細は、「SDP Architecture Guide」[6]を参照してください。

下図の青線は、MITM攻撃への対策として、mTLSやInternet Key Exchange（IKE）などの相互認証された暗号プロトコルによって保護されたネットワーク接続を示します。灰色の線は、ネイティブなアプリケーションプロトコルを示しており、暗号化されている場合もあれば、されていない場合もあります。図中では、簡略化のため、SDPコントローラを省略しています。

いくつかの配備モデルでは、SDPゲートウェイを使用しています。SDPゲートウェイは、SDP Accepting Hostとして動作するソフトウェアエージェントであり、保護対象のサービスの隔離とアクセス制御を提供します。

6 CSA Software-Defined Perimeter Architecture Guide, May 2019, page 14～18



1つ以上のサーバーがSDPゲートウェイの背後で保護される場合、IHとAH （ゲートウェイ）の間の接続は、下層にあるネットワークトポロジーに関係なく保護されます。Client-to-Gateway モデルでは、ゲートウェイはリモートからアクセス可能ですが隠蔽されており、安全な境界を提供します。このモデルでは、保護対象のサーバーに変更は必要ありません。

組織がエンドツーエンドの通信を保護する必要がある場合、Client-to-Serverモデルでは、サービスとAHを単一のホストに統合します。このモデルでは、サーバーは安全な境界内に隠されています。このモデルでは、AHとして動作するSDPソフトウェアがサーバーに必要です。

Server-to-Server モデルでは、下層にあるネットワークに関係なく、サーバー間のすべての接続が暗号化されていることを保証します。このモデルでは、すべてのサーバーが安全な境界内に隠されます。

VoIP、チャット、ビデオ会議サービスのように、ピアツーピアのトラフィックが中継サーバーを経由する場合があります。Client-to-Server-to-Clientモデルでは、サーバーが安全な境界内に隠されます。

Client-to-Gateway-to-Clientモデルは、Client-to-Server-to-Clientモデルのバリエーションです。このモデルは、アクセスポリシーを適用しながら、クライアントが互いに直接接続することを必要とするピアツーピアのネットワークプロトコルをサポートします。このモデルでは、ゲートウェイは境界の中に隠されます。

Gateway-to-Gatewayモデルは、SDP Specification 1.0に含まれていませんでした。
このモデルは、特定のIoT環境に適しています。このモデルでは、ゲートウェイは境界内に隠されます。

図3 - SDPの配備モデル

SDPのワークフロー

一般的に、SDPコンポーネントのワークフローは、オンボーディングワークフロー（各コンポーネントに個別のフロー）と、アクセスワークフロー（コンポーネント間で調整される）といった、2種類に分かれます。定義によると、各SDPコンポーネントは一度のオンボーディングワークフローを経て、その後、複数のアクセスワークフローに参加します。

これらのワークフローは代表的なものであり、実装やSDP配備モデルによって、具体的な内容は異なります。

コントローラのオンボーディングワークフロー

SDPシステムには、1つ以上のコントローラが必要です。オンボーディングフローが成功するためには、少なくとも1つのコントローラが常に使用可能でなければなりません。実装によっては、アクセスワークフローを成功させるために、アクティブなコントローラが必要な場合もあります。コントローラは、他のSDPコンポーネントが動作するすべての場所からアクセスできる必要があります。そのため、許可されたユーザーやデバイスに限られますが、通常、インターネットからグローバルにアクセスできます。

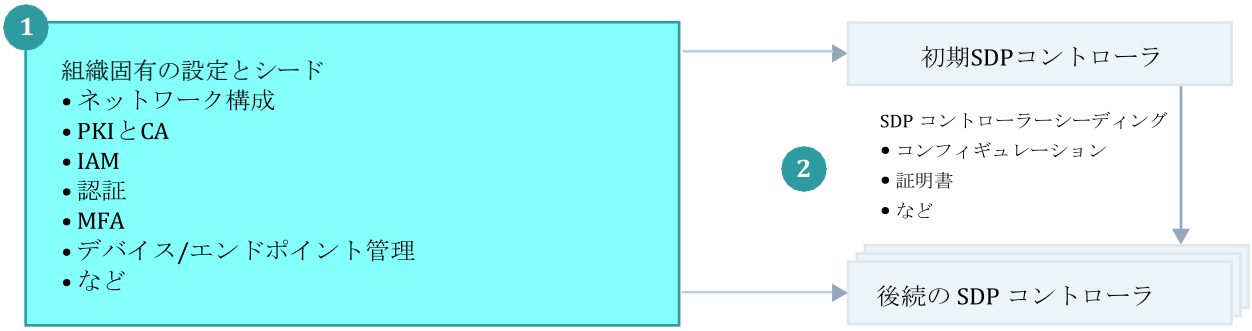


図4 - Controller Onboarding ワークフロー

図4は、ワークフローを示します。初期のSDPコントローラ（プライマリ）が起動し、適切な認証および認可のサービス（PKI認証局サービス、デバイス認証、地理情報、SAML、OpenID、OAuth、LDAP、Kerberos、多要素認証、などのサービス）と接続されます。SDPコントローラは、他のSDPコンポーネントがすぐに利用できるように、継続的かつ無期限に稼働します。後続のコントローラをオンラインにして、組織固有の同じ構成を行い、最初のコントローラからシード情報の引き継いでオンボードすることもできます。

多くのSDPの実装は、ロードバランシングと冗長化の目的で、コントローラのクラスタ構成をサポートします。各SDPの実装は、後続のコントローラをクラスタ内の他のコントローラに接続し、適切な状態を共有する仕組みをサポートする必要があります。
このメカニズムは実装に依存するため、本仕様の範囲外です。

Accepting Host (AH) のオンボーディングワークフロー

すべてのSDPシステムには、1つ以上のAHが必要です。これらのAHは、前述したSDP配備モデルのいずれかを使用して配置することができます。すなわち、スタンドアロンのゲートウェイを使用したり、リソースやワークロードを提供するサーバーの一部 として配置されたりします。

SDPの実装では、AHは長期間稼働しても、一時的であっても構いません。スタンドアロンゲートウェイのAHは、数ヶ月または数年など長期間稼働することがあります。一方、負荷に応じて拡大または縮小するゲートウェイの動的なクラスタでは、短時間の稼働になる場合もあります。

個々のサーバー（ワークロード）内に配置されたAHも、長時間または短時間の稼働となる可能性があります。この場合、AHの稼働時間は、AHが属するサーバーインスタンスの稼働時間と連動します。サーバーインスタンスは、従来のウェブサーバーやアプリケーションサーバーのように長期間稼働する場合もあれば、DevOpsインフラの一部のように短期間になる場合もあります。

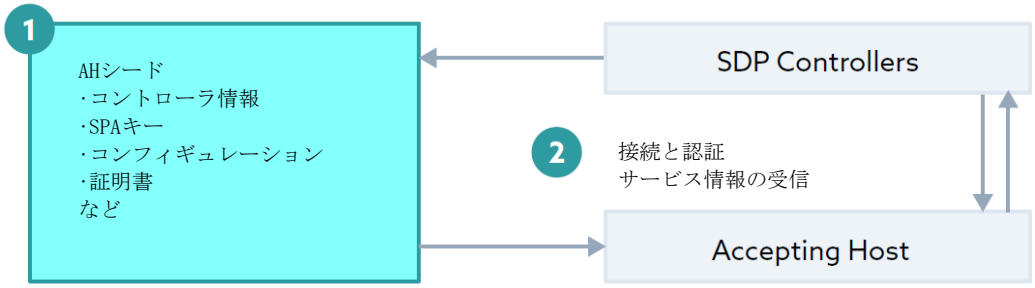


図5 - Accepting Hostのワークフロー

図5は、AHのオンボーディングワークフローを示します。 AHがサービスを開始するとき、AHはSDP内の1つ以上のコントローラに接続し、認証する必要があります。AHがオンボードされると、SPAパケットを受信し、認可されたIHからのアクセスを処理する準備ができます。

各SDPの実装は、AHがクラスタ内のコントローラに接続するメカニズムをサポートする必要があります。このメカニズムは実装に依存するため、この仕様のスコープ外です。同様に、多くのSDP実装では、ロードバランスと冗長化の目的で、一般的にゲートウェイ配備モデルのひとつを使いながら、実装に依存したAHのクラスタをサポートします。

Initiating Host (IH)のオンボーディングワークフロー

Initiating Host（IH）は、ユーザー機器、またはユーザーを持たないシステム（IOT機器やIHとして機能するサーバーなど）です。SDP システムのすべてのコンポーネントと同様に、IH は図 6 に示すようにオンボー ドする必要があります。このワークフローでは、コントローラへの接続に必要な初期情報が設定されます。これには、ネットワーク情報 （ホスト名または IP アドレス）と、必要な共有シークレット （SPA キー、証明書など）が含まれます。このメカニズムは実装に依存しており、この仕様の範囲外です。一般的に、これには企業のID管理プロバイダを使用する必要があり、ユーザーデバイスの場合、ユーザー認証はIHオンボーディングワークフロー内のステップであることが多くなります。

IHは一度だけオンボーディングする必要があり、その後、*アクセスワークフロー*を開始することができます。

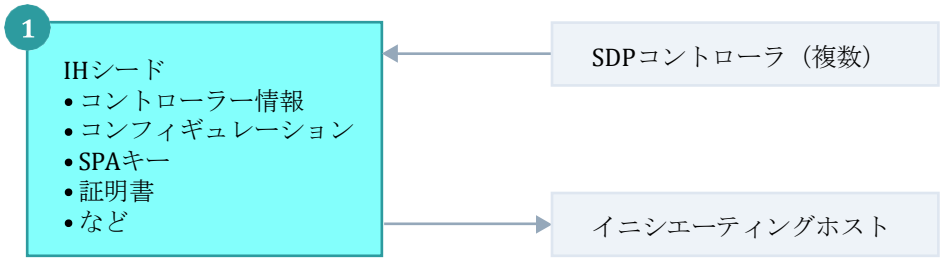


図6 - Initiating Hostホストオンボーディングワークフロー

アクセスワークフロー

IHは、表1および図7に示す手順で、SDPシステムで保護されたワークロード（リソース）に接続するための*アクセスワークフロー*を開始します。アクセスフローは、SDPのすべてのコンポーネントを含みます。IH、コントローラ、およびAHです。

ステップ	アクセスワークフロー
1	オンボーディングされたIHがオンラインに戻ると（デバイスの再起動後、またはユーザーが接続を開始した場合など）、SDPコントローラに接続し、認証を行います。
2	IHを（場合によっては対応するIDプロバイダで）認証した後、SDPコントローラは、 IHの通信が許可されているサービスのリストを（AHによって）決定します。 コントローラはまだこのリストをIHに伝えていないので、ステップ3以降を待つ必要があります。
3	SDPコントローラは、AHにIHからの通信と、双方向の暗号化通信に必要なユーザー、 デバイス、サービス間の接続を定義するすべての情報を受け入れるように指示します。
4	SDPコントローラは、認可されたAHとサービスのリスト、および双方向の暗号化通信 に必要なオプション情報をIHに提供します。
5	IHはSPAプロトコルを用いて、SPAの情報を検証する各認証AHとの接続を開始し、実施します。その後、IHはそれらのAHと双方向TLS接続を作成します。

表1：アクセスワークフロー

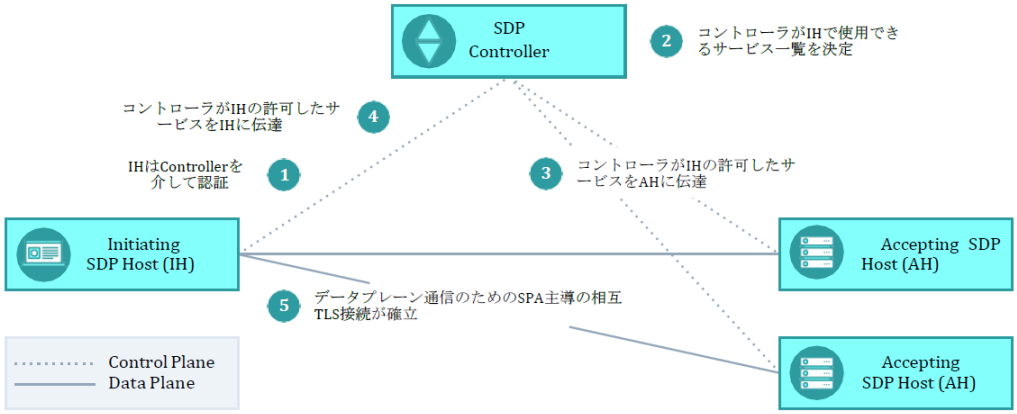


図7

IHオンボーディングのワークフロー例

SDPコントローラ、IH、AH、ユーザーのオンボーディングは、図2に示す配備モデルや実装の仕様によって異なります。SDPシステムの構築・運用は、API（Application Programming Interface）や管理用UIを介して行われます。

上記のワークフローで説明したように、SDPシステムが配備され、構成され、SDPコントローラとAHがオンラインになります。

以下の流れは、外部のIdP（Identity Provider）で認証するIHのユーザーの場合のオンボーディングワークフローになります。

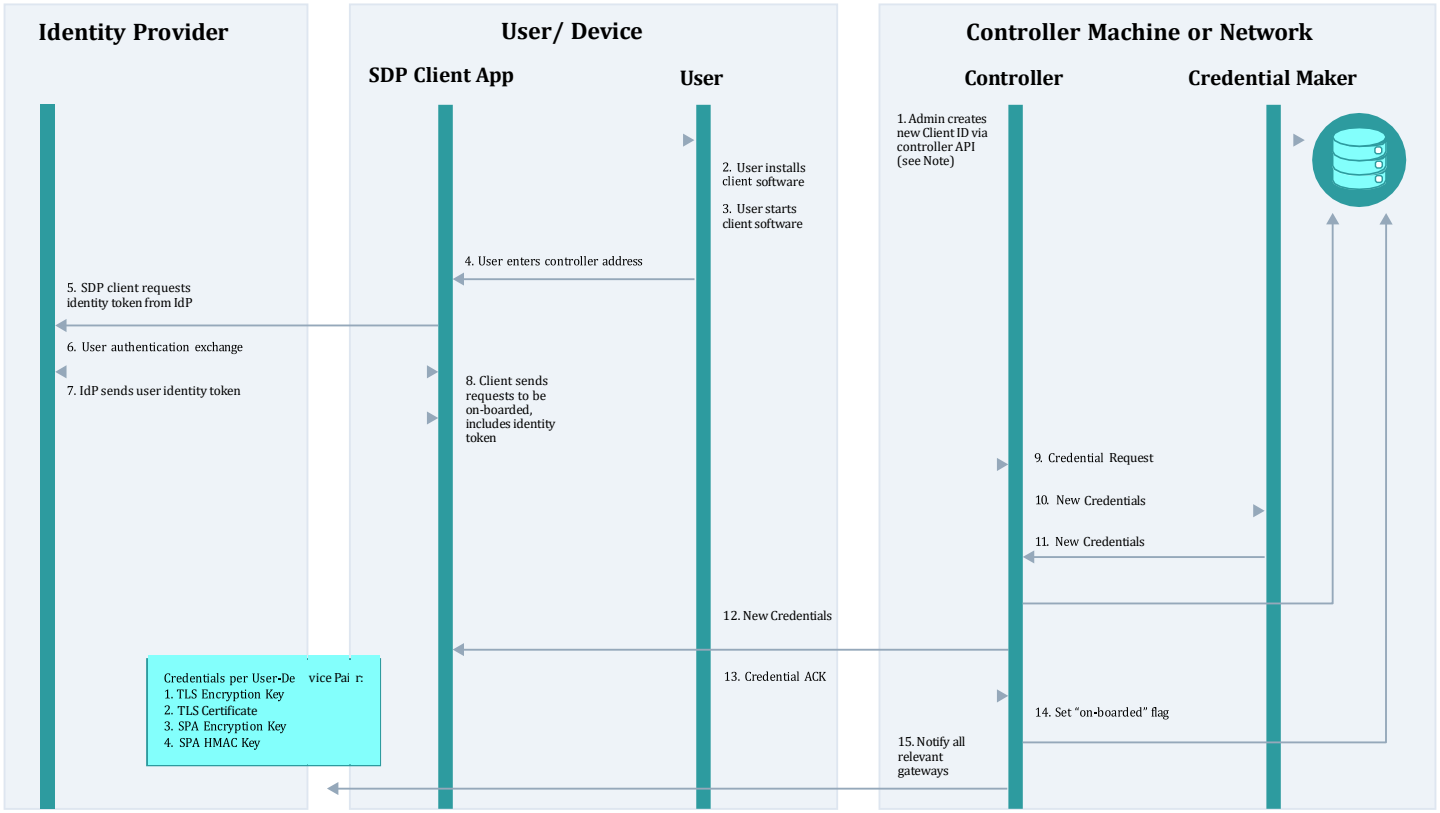


図8 - オンボーディング(IdPシナリオ)

7 SDP Open Source Reference Implementation - https://www.waverleylabs.com/open-source-sdp/ に基づいています。

注 - この例では、ユーザー登録のために IdP情報を使用しています。登録システムは、登録プロセスの一部として、MFAやデバイス属性の検証（企業に配備された証明書やエンドポイント管理ソフトウェアなど）など、複数の認証要素を含めることが可能です。この例では、実装に依存したノードへの鍵素材（SPA キーや他の共有シークレットなど）の配布も省略しています。たとえば、オープンソースのSDP実装では、この鍵素材を生成し、企業が何らかの安全な方法でSDPのコンポーネントに安全に送信することも可能となります。

このサンプルワークフローでは、IH はクライアント （図では SDP Client） を使用して IDP (Identity Provider) とネゴシエートします。一部の SDP モデルでは、SDP コントローラは IdP と信頼関係を結びます。たとえば、 クライアントから転送された SAML トークンを検証するためです。

Single Packet Authorization（SPA）

SDPの基本原則の1つは、ワークロード（リソース）だけでなく、SDPインフラ自体も、権限のないエンティティがアクセスできないようにする必要があることです。これは、エンティティがSDPコンポーネントに接続する際に、暗号手法を用いて認証されることを義務付けることで実施されます。未承認のエンティティはSDPコンポーネントとのネットワーク接続を確立できないため、脆弱性を悪用したり、ログインを総当り的に試みたり、盗まれたユーザー認証情報を利用したりすることができません。これは、インターネット上のすべての悪意ある行為にさらされることが多いVPNなどの従来のリモートアクセスソリューションとは対照的です。[8]

SDPがこのために使用するメカニズムが、Single Packet Authorization (SPA)です。SPAのSDPバージョンは、RFC 4226 HMACベースのワンタイムパスワード 「HOTP」[9] に基づいており、以下のように、SPAパケットに含まれます。

SDPは仕様上、コントローラやAHへのアクセスを保護するためにSPAを使用することを目的としていますが、適切なアーキテクチャで動作する信頼できる代替手段もあります。以下の「*SPAの代替手段*」のセクションを参照してください。

SPAがSDPの中で実現する重要な原則は以下の通りです。

- **SDPシステムコンポーネントを隠蔽：**コントローラもAHも、リモートシステムからの接続試行に対して、以下のように SDPシステムに対して有効 なSPAパケットが提供されるまで応答しません。具体的には、UDPベースのSPAの場合は特に、ホストはTCP SYNに応答しないことで、 潜在的な攻撃者に情報を開示することを回避することができます。（デフォルトドロップファイアウォールは、この実装方法の一例です）。これは、スタンドアロンAH（SDPゲートウェイ）と、サーバー/ワークロードの 論理的な一部であるAHの両方に当てはまります。
- **TLSのサービス拒否攻撃を軽減：**インターネットに面した HTTPS（TLSを使用）プロトコルを実行しているサーバーは、サービス拒否（DoS）や分散型サービス拒否（DDoS）攻撃を非常に受けやすくなっています。SPAでは、サーバーが不正な接続をすばやく拒否できるため、これらの攻撃を軽減できます。

8 公正を期すために、インフラを隠す商用およびオープンソースのVPN的ソリューションがいくつかあります。Wireguardは、そのようなオープンな例の1つです。

9 https://tools.ietf.org/html/rfc4226 をご覧ください。

TCP または TLS 接続を確立するため、DoS 攻撃中、およびそれにもかかわらず許可された接続を可能にします。

- **攻撃の検出：**コントローラまたは他のホストからAHへの最初のパケットは、SPAパケットでなければなりません。それ以外のパケットをAHが受信した場合、攻撃と見なす必要があります。そのため、SPAを使用することで、SDPは1つの悪意のあるパケットに基づいて攻撃を判断することができます。

設計上、受信するSPAパケットの検証は計算が軽く、DDoS攻撃に対するSDPシステムの回復力が向上することに注意してください（SDP Cloud Security Alliance, *SDP as a DDoS Defense Mechanism*ホワイトペーパー[10]で言及されています）。

SPA は、IH-Controller、AH-Controller、IH-AH の通信を開始します。SPAパケットは、選択された実装に応じて、UDPまたはTCPプロトコルを使用して開始されます。

UDPベースのSPAは、SPAで保護されたサーバーに上記のすべてのセキュリティ上の利点を提供します。TCP ベースの SPA は、UDP ベースの SPA よりも少ないものの、これらの利点のいくつかを得ることができます。具体的には、TCP SPAを使用するSDPコンポーネントは、すべてのリモートユーザー(悪意のある可能性がある)にオープンポートを公開するため、サーバー（SDPゲートウェイ、AH）は隠蔽されません。また、サーバーは部分的にDDoS攻撃の対象になります。
-- リモートIPアドレスからのTCP接続を許可し、TLS接続を作成する前にSPAの検証を行います。TCP接続はTLS接続よりもはるかに少ないリソースしか必要としませんが、サーバーのリソースを消費し、サーバーをDDoSの危険にさらすことになります。最後に、TCP SPAを使用すると、TCPは無効なSPAパケットに基づく攻撃をサーバーが検出することを許可しますが、TCP接続が確立された後でなければならず、それによってサーバーのリソースが消費されます。

サービス露出を最小化するためのもう一つの考慮点は、AHとコントローラのDNS列記です。IP経由で直接接続したり、プライベートDNSサービスのみを経由すると、攻撃者からのサービスの可視性を低下させる可能性があります。つまり、SDPコンポーネントのパブリックDNSエントリ（`controller1.sdp.mycompany.com` など）の存在自体が、悪意のある行為者の攻撃対象 になる可能性があります。たとえば、SDPインフラストラクチャに対する大規模なDDoS攻撃 などです。

以下の推奨SPAメッセージフォーマットは、SDP仕様のv1以降、プロトコルのセキュリティと耐障害性を向上させるために更新されていることに注意してください。

SPAメッセージのフォーマット

SPAメッセージの形式はSDP実装によって異なる可能性がありますが、すべてのSDPシステムは、コンポーネント間の接続を開始するためのメカニズムとしてSPAをサポートする必要があります。SPAを使用するには、SPAパケットの作成者と受信者が信頼の基点を共有する必要があります。これは、有効なSPAパケットを構築するために、各SPAパケットに共有のシークレットが必要だからです。この信頼の基点を確立するために -- つまり、共有シークレットをSDPコンポーネントに安全に伝達する方法は、- 実装に依存しており、本仕様の範囲外になります。一般的に、この情報は、IHとAHの登録プロセスに含まれます。

10 https://cloudsecurityalliance.org/artifacts/software-defined-perimeter-as-a-ddos-prevention-mechanism/

ClientID	ユーザーとデバイスのペアごとに割り当てられる256ビットの数値識別子。このフィールドは、パケットを送信しているユーザー、デバイス、または論理グループを区別するために使用されます。
Nonce	16ビットのランダムデータフィールドにより、SPAパケットの再利用を回避し、リプレイ攻撃を防止します。
Timestamp	短い有効期間（例えば15～30秒）を確保することで、古くなったSPAパケットのサービスを防止します。また、受信者に必要なリプレイ検出のキャッシュを減らす仕組みも提供します。
Source IP Address	initiating hostの*一般に公開されている*IPアドレス。これは、途中で簡単に変更されるパケットヘッダ中のソースIPアドレスに受け入れホストが依存しないように含まれるものです。IHは、AHがパケットの発信元として使用するためのIPアドレスを取得できなければなりません。
Message Type	このフィールドはオプションであり、接続確立後にIHからどのような種類のメッセージを期待するかを受信者に知らせるために使用することができます。
Message String	このフィールドはオプションであり、Message Type フィールドに依存します。例えば、このフィールドは、もし接続時に分かっていれば、IHが要求するサービスを指定するために使われるかもしれません。
HOTP	このハッシュ化されたワンタイムパスワード（ハッシュOTP）は、共有シークレットに基づいて、RFC4226で説明されているアルゴリズムで生成されます。OTP の使用は、SPA パケットの真正性を確立するために必要です。SPA パケットの真正性を提供するという包括的な目標があれば、他の OTP アルゴリズムに置き換えることができます。
HMAC	上記のすべてのフィールドを対象として計算されます。アルゴリズムは、SHA256（推奨）、SHA384、SHA512、SM3、Equihash、またはその他の効率的で堅牢なアルゴリズムから選択することができます。HMACは、共有（シークレット）シードを使用して計算されます。HMACはメッセージのすべての先行フィールドに対して計算され、AHがメッセージの完全性を検証するために使用されます。HMACの検証は計算上軽量であるため、DoS攻撃に対する回復力を提供するために使用することができます。無効なHMACを持つSPAパケットは直ちに破棄されます。

表2 SPAメッセージスキーム

他のSPAオプションでは、例えばIHの秘密鍵（否認防止用）やAHの公開鍵（機密保持用）を使った追加の暗号化が可能であることに注意してください。しかし、非対称暗号化は計算量が多く、AH を DoS 攻撃に耐えられるようにするため、受信者はより軽量な検証メカニズム（単純な HMAC など）の後にのみ使用するようにすべきです。

安全な自己完結型コネクションレス型メッセージ伝送プロトコルとしてのSPA

SDPシステム内でのSPAの興味深い「サイド」ユースケースの1つは、SPAパケットの接続を開始するための手段としてだけでなく、リモートエレメントからデータを送信するための手段として使用することができることです。SPAパケットは共有シークレットに基づいているため、受信者は、その中に含まれるデータが有効なSDPクライアントから発行されたものであると信頼することができます。

SPAシードが特定のクライアントに固有である場合（SPAパケット内のClientIDで識別）、Message Stringフィールドを使用して、クライアントが意味のあるデータを送信することができます。この場合、さらなる処理やポリシー評価は不要であり、TCPやTLS接続を確立する必要もありません。

これは、例えば、少量のデータを定期的に送信する必要がある一連の分散型のIoTセンサーに有効です。SPAパケットにデータを埋め込むことで、TCPおよびTLS接続のオーバーヘッドを発生させずに、このようなデバイスを実現できます。もちろん、このメカニズムにはいくつかの欠点があります。受信者(AH)はこのデータを期待しているはずで、これは一方向の送信なので、送信者はSPAパケット送信でデータが実際に受信されたことを確認することができません。それでも、データが重要/機微でない場合に限り、これはある種の環境においてSPAを適用する有用な方法となり得ます。

SPAの代替品

SDPアーキテクチャは柔軟性を念頭に置いて設計されており、今回の仕様改訂では、作成される可能性のあるアプローチや配備モデルの多様性を取り込んでいます。たとえば、SDPは6つの配備モデルを定義しています。それぞれのモデルは、異なる問題の解決に適しており、異なる方法で実装され、異なるトレードオフを生み出す可能性があります。SPAは、インフラストラクチャの隠蔽、DDoSに対するレジリエンスの提供というSDPの主要な原則を達成するための健全で安全なメカニズムです。
また、SPAはSDPを自己完結型のシステムにすることができるという利点もあります。IHは一度シードを与えると、外部システムに依存することなく、安全かつ確実にコントローラやAHとの暗号化通信を確立することができます。

しかし、SPAは、システムがこれらの目標を達成するための唯一のメカニズムではありません。ここで定義した原則を実現する代替アプローチもあり、SDPとゼロトラストの原則をサポートするシステムの一部とすることも可能です。たとえば、SPA を使用しない SDP システムでは、グローバルにアクセス可能なエンタープライズ IdPを使用し、SDP コントローラとAH に制御チャネルを提供することができます。
このモデルでは、IdPに対するIHの認証がトリガーとなり、SDPコントローラとAHにIHの認証が成功し、IHのパブリックIPアドレスからの即時接続を期待するように通知する制御プレーンメッセージが発行されます。IHは、コントローラおよびAHへのTCP接続を確立することができます。TCP接続の後には、もちろん（相互認証TLS（mTLS））および以下に定義する他のセキュリティのレイヤーが続くことになります。

もちろん、どのようなシステムやアーキテクチャにもトレードオフがあります。そのアプローチが上述の3つの原則を達成する限り、SPAに代わるものを使用するシステムは、健全で価値のあるSDPシステムの一部となり得るのです。

コンポーネント間のトランスポートレイヤの相互認証

SDPは、接続確立プロセスを通じて、複数のレイヤーの検証を必要とします。最初の段階は、前述のとおりSPAです。このセクションの主題である次のステップは、分散SDPコンポーネント間の安全な暗号化接続の確立の一部としての、相互認証の要件についてです。デバイスとユーザーの検証を含む追加の手順については、以下で説明します。

コンポーネントがSPAを使用して認証および認可された後、SDPシステムの主要コンポーネント間の接続には、TLSまたはInternet Key Exchange (IKE)などの他の代替手段を使用する必要があります。
デバイスをSDPの認可メンバーとして検証するために、相互認証付きのIKE（Exchange）を使用します。具体的には、開始ホストと受け入れホスト（IH-AH）、開始ホストとコントローラ（IH-Controller）、コントローラとAHの接続は、相互認証を使用する必要があります。TLSはTCP上でもUDP上でもサポートされていることに注意してください（その場合、Datagram Transport Layer Security、またはDTLSと呼ばれます）。SDPシステムではどちらも許容されます。

相互認証をサポートしない脆弱な暗号スイートやプロトコルは不適切であり、安全とは言えません。強力な暗号とプロトコルは、各コンポーネントが信頼できる機関から発行された有効な秘密鍵を含むことを保証し、双方向で署名された証明書を検証することにより、中間者攻撃（MITM）[11]の可能性を大幅に減少させることができます。

これらのコンポーネントのルート証明書は、企業のPKIシステムまたはSDP固有のCAでなければなりません。消費者向けブラウザに関連する発行済み証明書または暗黙的に信頼された証明書に依存してはなりません。これは、攻撃者が侵害された認証局からの証明書を偽造するなりすまし攻撃の対象になります。SDP は、オンライン証明書ステータスプロトコル（OCSP）[12] または他のメカニズムを使用するなどして、失効した証明書を効率的に検出できるようにするスキームを実装する必要があります。

SDPシステムがどのようなトランスポートレイヤーアプローチを使用しても、MITM TLSダウングレードプロトコル攻撃(mTLSは回避)など、潜在的な攻撃に対してレジリエンスがなければなりません。

デバイスの検証

相互トランスポートレイヤ認証は、SDPへのアクセスを要求するデバイスが、有効期限内で取り消されていない秘密鍵を所有していることを証明しますが、デバイス がセキュリティ要件や構成要件を満たしているかどうかは検証しません。

デバイス検証の目的は、デバイスがセキュリティ要件を満たしていることを証明することです。企業環境では、コントローラは最も管理された環境に存在するため、信頼できるデバイスであると仮定されることに注意してください。SDPゲートウェイも、SDPを運用する企業の管理下にある場合、信頼できるコンポーネントであると想定されます。つまり、これらのコンポーネントは管理された環境で実行され、企業の変更管理制御と配備プロセスの対象となり、SDPシステムに意図的にオンボードされる必要があります。コントローラおよびゲートウェイコンポーネントの場合、デバイスの検証はオプションであると考えるべきです。

一方、ユーザーデバイス（IH）には、デバイスの検証が必要です。デバイスバリデーションは、クレデンシャル盗難とそれに起因するなりすまし攻撃を軽減します[13]。ユーザーデバイスは、デバイス検証とセキュリティポリシー全体の要件を満たした後、SDPコントローラへの接続が許可されます。このプロセスにより、攻撃者がmTLS認証用の正しい秘密鍵を所有していたとしても、AHの上または後ろのサービスにアクセスできないことが保証されます。

11 https://wott.io/blog/tutorials/2019/09/09/what-is-mtls
12 https://tools.ietf.org/html/rfc6960
13 秘密鍵および SPA 秘密は安全に保管し、定期的にローテーションする必要があります。これは今後の課題です。

パケットは、SDPを介して認証および認可されない限り、どのようなデバイスからもドロップされます。したがって、このプロセスは、未認可のデバイスからの受信パケットを防ぐことができます。デバイス検証のプロトコルや詳細は、企業や製品に固有であるため、SDP仕様の範囲外ですが、今後の課題として注目される分野です。

SDPシステムは、企業のデバイス管理／エンドポイント管理システムと統合する機能をサポートし、デバイスのセキュリティポスチャーチェックをデバイス検証プロセスに含める必要があります。さらに、SDPシステムは、SDPがユーザーデバイス上でローカルにソフトウェアを実行している環境において、ローカルデバイスのセキュリティポスチャーチェックを実行する機能をサポートする必要があります。たとえば、SDPクライアントは、ユーザーデバイスに企業が発行した有効な証明書が含まれていることを確認できます。この証明書は、mTLS認証に使用される可能性があります。また、デバイスで承認されたアンチウィルスコンポーネントが実行されていることを検証することもできます。これらの側面は、SDPがサポートする全体的なゼロトラストアプローチに関連していることに注意してください。また、ここでの「デバイス」は、サーバーを指すこともあります。サーバーは、 ユーザーデバイスと同じ方法で検証することができ、検証する必要があります。

SDPとIoTデバイス

NIST 800-207 のゼロトラストの中核的な考え方の 2 つは、システムが「すべてのデータソースとコンピューティング サービス」、および「すべての通信」[14]に対するアクセスを保護しなければならないとしており、これには IoT デバイスが含まれる必要があります。IoT は、ほとんどのIT環境に存在するデバイスを含む広義のIoTと考えています（例えば、プリンタ、IPカメラ、VoIPフォン）、環境・医療用センサーや産業用制御システムなど、より特殊なデバイスも含まれます。

これらのデバイスがオープンであり、通常のSDP Initiating Hostとして動作する任意の完全な機能のソフトウェアのインストールと操作をサポートできる場合、IHとして動作する通常のホストと何ら変わりなく扱うことが可能です。ここでは、3つのカテゴリの1つに該当するデバイスに限定して議論します。クローズド（非拡張型）デバイス、つまり追加ソフトウェアをインストールする、汎用コンピュータ（Arduinoなど）であるがコンピュート/メモリ/ストレージの容量が限られている、または、SDPクライアントソフトウェアを配備しないことを選択した上ですべての機能を持った汎用システムである場合、企業はこのデバイスを使用できません。どのような場合でも、これらのデバイスはネットワーク接続されており、これらのデバイスへのアクセスは、SDPアーキテクチャとポリシーモデルに含まれる必要があります。

クローズドデバイスがSDPシステムに含まれるためには、IoTデバイスのネットワークトラフィックを取得し、それをSDPシステムに仲介する「アップストリーム」ネットワークデバイス（「ブローカー」または「コネクタ」）が必要です。SDPブローカーは、IoTデバイスに代わって論理的なIHとして機能します。このブローカーは、IoTデバイスへのトラフィックとIoTデバイスからのトラフィックの両方を制御できる必要があることに注意してください。

容量制限のあるデバイスでは、SDPソフトウェアの一部を実行できますが、フル機能のIHとして動作させることはできませんし、そうすべきでもありません。SDPは、これらのデバイスから、またはデバイスへの通信のための代替モデルに対してオープンです。たとえば、低電力、低容量の環境センサーの配列は、データを安全に送信するために有効なSPAパケットのみを生成する軽量のSDPクライアントを実行することができます。TLS 接続の確立や認証が不要なため、非常に計算量の少ない方法で、データを安全に伝送することができます[15]。

14 NIST Special Publication 800-207「Zero Trust Architecture」（6ページ）を参照。

15 これはメッセージの完全性を提供しますが、データは転送中に暗号化されないため、プライバシーは提供しません（ただし、暗号化することは可能です）。

これらのアイデアは、ゼロトラストとIoTに関する今後の研究の良い出発点であり、この分野におけるさらなる研究と出版を期待しています。

アクセスポリシー

ゼロトラストは、保護されたリソースへのアクセスを制御する*アクセスポリシー*の考え方にしっかりと軸足を置いています-- ゼロトラストの主要な概念である*ポリシー決定*ポイントと*ポリシー実施*ポイントを思い出してください。これは、NISTの*Zero Trust Architecture文書*の中核的な考え方の1つで、明確に書かれています。「リソースへのアクセスは、クライアントのアイデンティティ、アプリケーション/サービス、要求している資産の観測可能な状態を含む動的ポリシーによって決定され、他の行動および環境属性を含むことがあります」[16]。また、Cybersecurity & Infrastructure Security Agency（CISA）のZero Trust Maturity Modelでは、*従来*の成熟度レベルでは、静的ポリシーから、上級レベルではクロスドメインの入力と出力を使い、*最適*レベルでは「自動/観測トリガーに基づく動的ポリシー」[17]になることを論じています。

しかし、ゼロトラストのアーキテクチャや企業要件が多種多様であるため、効果的でありながら広く適用できるポリシーモデルのフレームワークと関連する語彙を定義することは困難です。たとえば、SDPだけで、6つの異なる配備モデルがあり、それぞれ実施機能がわずかに異なっています。このような幅の広さの結果、初期のゼロトラスト文書は、しばしばポリシーのトピックについて沈黙していました；たとえば、このSDP仕様書のバージョン1では、政策モデルについて議論も定義もしませんでした。新しいゼロトラストモデルは、ポリシーにどのようにアプローチするかについて、いくつかのガイダンスを提供しています。

NIST 800-207 は、図 9 に概念的に描かれた、*信頼アルゴリズム*を評価する*ポリシーエンジン*の概念を導入しています。

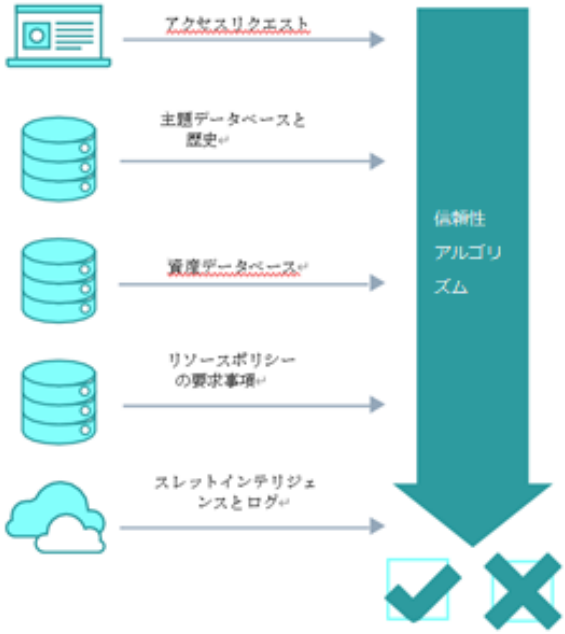


図9：NIST800-207のトラストアルゴリズムの図。

NIST 800-207は、リソースポリシーの要件について述べているところで、ポリシーモデルについて触れています：

> "この一連のポリシーは、ユーザーIDと属性データベース[SP800-63]を補完し、リソースへのアクセスのための最小限の要件を定義します。要件には、MFA ネットワークの場所（例：海外 IP アドレスからのアクセスを拒否する）、データの機密性、資産構成の要求などの認証保証レベルを含めることができます。これらの要件は、データカストディアン（すなわち、データの責任者）と、データを利用するビジネスプロセスの責任者（すなわち、ミッションの責任者）の両方によって策定されるべきです。"

別途配布される鍵で）。*（訳注：これは、上記の本文に続くものと思われるが、原文に基づいてここに翻訳した）*

16 NIST Special Publication 800-207「Zero Trust Architecture」（6ページ）を参照。

17 US Cybersecurity and Infrastructure Security Agency Cybersecurity Division, Zero Trust Maturity Model, Pre-decisional Draft, June 2021, Version 1.0, Page 5.

CISAのゼロトラスト成熟度モデルは、ポリシーに直接言及していませんが、ポリシーで表現可能な異なる成熟度レベルのシステム能力を例示することで、間接的にそれを実現しています。

普遍的に適用可能なゼロトラストポリシーモデルの構造と語彙を定義することは困難です。これは、セキュリティ哲学としてのゼロトラストの性質に起因するものです。SDPアーキテクチャはより具体的であり、SDPの配備モデルによって違いはありますが、構造化されたポリシーモデルに適していると考えています。たとえば、デバイスに企業が発行した証明書が含まれていることを確認することを要求するデバイスポスチャチェックポリシーを考えてみましょう。これは、ユーザーのデバイス上でローカルエージェントが実行されているシステムでは簡単に実施できますが、エージェントレスのシナリオではより困難です。[18] このような現実世界の制約により、業界の研究およびガイダンス文書（必然的に実装に中立でなければならない）は、具体的で詳細なポリシーモデルではなく、より一般的なポリシー側面（例：ユーザーデバイスが企業のセキュリティポスチャーチェックを満たすようにするなど）に限定されることになります。

ゼロトラストポリシーモデルは、このv2 SDP仕様の範囲外ですが、SDPおよびZTワーキンググループ内だけでなく、業界全体で将来的に取り組むべき興味深い分野です。このトピックに関する今後の協力や出版を期待しています。

SDPプロトコル

SDPプロトコルは、AH-コントローラプロトコル、IH-コントローラプロトコル、IH-AHプロトコル、ロギングの4つのセクションから構成されており、以下で詳しく説明します。本書で示すSDPプロトコルは、SDPコンポーネント間のSPAを使用した通信を示すものですが、これは規範となるものではありません。本書の目標は、動作する代表的なシステムを示すことであり、固定または標準化されたメッセージ形式を規定することではありません。

また、このプロトコルには、前述のようにコンポーネントやデバイスのオンボーディングのためのメッセージやデータフローは含まれないことに注意してください。

なお、ここでは、わかりやすくするために、タイムアウト、リトライ、その他のエラー処理を規定していません。また、IHのログイン処理では、コントローラが、例えばIHを認証するための企業ID管理システムなど、追加の外部呼び出しを行うことがあります。最後に、以下の例では、コンポーネント間でTCPおよびTLS接続を使用していることに注意してください。先に述べたように、コネクションレスのUDPベースのプロトコル（すなわちDTLS）を使用することも可能ですが、わかりやすくするために以下の例では記載していません。

18　また、OSによってはエージェントがないと実現できない場合もあります

AH-コントローラプロトコル

AH to コントローラシーケンス図

AH（Accepting Host）がコントローラ（Controller）へ接続するためのプロトコルシーケンスを図10に示します。UDPベースのSPAパケットが最初のパケットとなり、SDPコントローラを不正なアクセスから保護します。

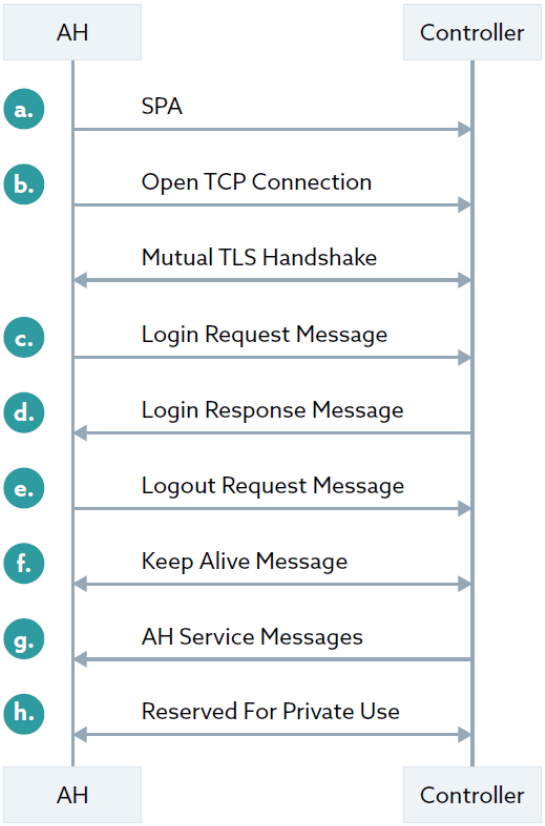


図10: Accepting HostがControllerに接続する様子

以下では、AHとコントローラの間でやり取りされる様々なメッセージやフォーマットを定義します。基本的なメッセージの形式は、次の通りです。

コマンド（8ビット）	コマンド指定データ（コマンド指定長）

a. SPA

SPAパケットは、AHがコントローラに接続を要求するために、前述したフォーマットに従って送信されます。

b. TCPコネクションと相互認証された通信路（mTLS）の確立

SPAパケットを送信後、IHはコントローラとの間でTCPコネクションの確立を試みます。コントローラは、SPAパケットが有効であると判断した場合、このTCPコネクションの確立を許可します。

その後、mTLS接続の確立に必要な相互認証が行われます。

UDPベースのDTLSの場合、UDPはコネクションレス型プロトコルであるため、これは論理的なコネクションとなります。

c. ログイン（SDP参加）要求メッセージ

ログイン要求メッセージは、AHが動作中であることを示し、アクティブなAHとしてSDPに参加することを要求するために、AHからコントローラに送信されます。AHのログイン要求には、AHがコントローラに対して自身を識別および認証するための証明書を含めることができます。

0x00	コマンド固有データなし

d. ログイン（SDP参加）応答メッセージ

ログイン応答メッセージは、ログイン要求が成功したかどうかを示し、成功した場合はAHセッションIDを提供するためにコントローラからAHへ送信されます。ただし、AHのログイン要求をコントローラが拒否することもありえます。これは、たとえば認証情報が無効であることが原因です。あるいは、コントローラがシステムのライセンスや同時接続数の制限などを実施していて、そのためにAHのログインを拒否する可能性もあります。いずれの場合も、接続が拒否されるケースでは、コントローラはAHからのTCPコネクションを切断します。

0x01	ステータスコード(16ビット)	AHセッションID（256ビット）、およびオプションでAH用に更新されたSPAキーとTLSキー。

ステータスコードは実装に依存します。AHセッションIDは、システムログ内で使用されます。例として、これを表すJSON仕様を以下に示します。

フォーマット	例
{ "session_id": <256ビット>, "credentials": ["spa_encryption_key": <64ビット>, "spa_hmac_key": <64ビット>, "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. ログアウト要求メッセージ

ログアウト要求メッセージは、 AHがコントローラに送信するもので、 AHが利用できなくなったことや、コントローラからの他のメッセージを受け入れることができなくなったことを示します。コントローラからの応答メッセージはなく、AHまたはコントローラがTLSおよびTCPコネクションを切断する必要があります。

0x02	コマンド固有データなし

f. キープアライブメッセージ

キープアライブメッセージは、AHまたはコントローラのどちらかが、まだアクティブであることを示すために送信されます。

0x03	コマンド固有データなし

g. AHサービスメッセージ

サービスメッセージは、このAHが一連の保護するサービスを示すために、コントローラから送信されます。以下の例では、サービスが固定IPアドレスで指定されていることに注意してください。これは、AHがSDPゲートウェイの場合の例です。代わりに、AHが解決する必要があるホスト名でサービスを指定することもできます。AHがサービスを実行しているホストの一部である場合（図1A参照）、AHがアクセスをコントロールしているサービスを特定するのに、IPアドレスではなくidまたはnameフィールドを使用することができます。

0x04	JSON形式のサービスの配列

データプレーンのJSON仕様の例を以下に示します。

フォーマット	例
{ "services": ["port": <Server port>, "id": <256ビット Service ID>, "address": <Server IP>, "name": <service name>, "type": <protocol type tag>] }	{"services": ["port": "443", "id": "123445678", "address": "10.2.1.123", "name": "Marketing Web App", "type": "HTTPS"] }

サービスメッセージは、TCPやUDPベースのサービス、あるいは他のプロトコル（ICMPなど）を使ったサービスを記述するために使われる可能性があることに注意してください。

h. ユーザー定義メッセージ

このコマンド（0xff）は、AH とコントローラ間で非標準なメッセージを送受信するために予約されているものです。

0xff	ユーザー指定

IH-Controllerプロトコル

IH（Initiating Host）-Controllerプロトコルは、ネットワークのルーティングとパケット配送を操作し、実装の詳細はトランスポートの種類（配送の信頼を保証するTCP や送信したままで到達を保証しないUDP など）に依存します。

IH to Controllerシーケンス図

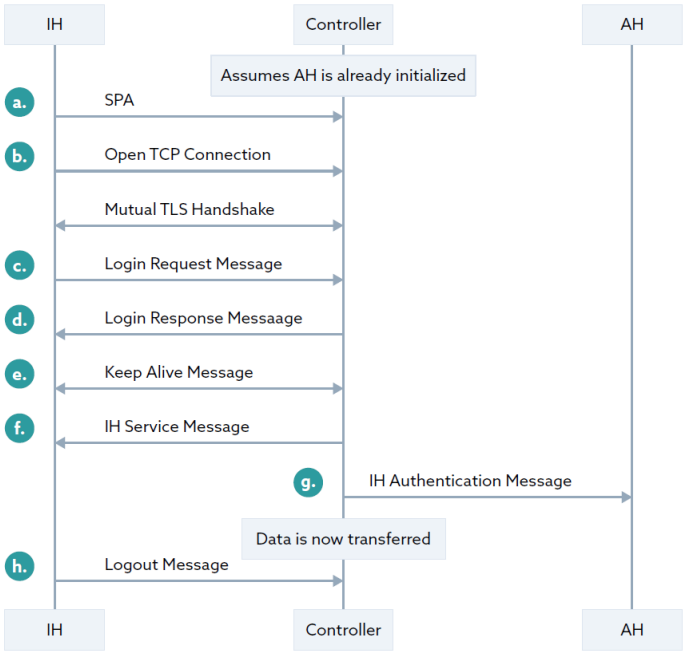


図11：Initiating HostがSDPコントローラに接続

以下では、IHとコントローラの間でやり取りされるさまざまなメッセージとその形式を定義します。基本的なプロトコルは以下の形式です。

コマンド（8ビット）	コマンド指定の長さのコマンド指定データ

a. SPA

SPAパケットは、IHからコントローラへ接続を要求するために、前述したフォーマットに従って送信されます。

b. TCPコネクションと相互認証された通信路（mTLS）の確立

SPAパケットの送信後、IHはコントローラとのTCPコネクションを確立します。コントローラは、SPAパケットが有効であると判断した場合、このTCPコネクションの確立を許可します。その後、mTLSコネクションを確立するために必要な相互認証が行われます。

UDPベースのDTLSの場合、UDPはコネクションレス型プロトコルであるため、これは論理的な接続となります。

c. ログイン（SDP参加）要求メッセージ

ログイン要求メッセージは、IHが使用可能であり、SDPに参加したいことを示すために、IHからコントローラに送信されます。IHのログイン要求には、IHを識別し、コントローラに対して認証するための証明書を含めることができることに注意してください。また、このログインリクエストは、本文書で前述したオンボーディングフローの後に発生することにも留意ください。このログイン要求は、セッションごとに発生します。たとえば、ユーザーが最初にデバイスの電源を入れたときに、1日に1回。

0x00	コマンド指定データなし

d. ログイン応答メッセージ

ログイン応答メッセージは、ログイン要求が成功したかどうかを示し、成功した場合はIHセッションIDを提供するためにコントローラからIHへ送信されます。ただし、コントローラが IH のログイン要求を拒否することもあります。これは、たとえば認証情報が無効だった場合などです。あるいは、コントローラがシステムのライセンスや同時接続数の制限を実施していて、そのために IH のログインを拒否している可能性もあります。

0x01	ステータスコード(16ビット)	IHセッションID（32ビット）、オプションでIHの更新されたSPAキーとTLSキー。

フォーマット	例
{ "session_id": <256ビット>, "credentials": ["spa_encryption_key": <64ビット>, "spa_hmac_key": <64ビット>, "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. キープアライブメッセージ

キープアライブメッセージは、IHまたはコントローラのどちらかが、まだアクティブであることを示すために送信されます。

0x03	コマンド指定データなし

f. IHサービスメッセージ

サービスメッセージはコントローラからIHへ送信され、IHに対して利用可能なサービスのリストと、それらを保護するAHのIPアドレスまたはホスト名を提供します。このメッセージには、IHがサービスに接続するために必要な情報が含まれていなければなりません。ホスト名/IPアドレスは、IHから直接到達可能なAHのものとなります。実際のサービスは、先の「導入モデル」の項で述べたように、AHとは異なるホスト/IPで動作している可能性があります。

サービスIDは、後にIHがAHと通信する際に、対象となるサービスを識別するために使用します。

0x06	JSON形式のサービスの配列

JSONの仕様例を以下に示します。

フォーマット	例
{"services":[{"address": <AH IP>, "id": <256ビット 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認証メッセージ

AH（Accepting Host）の役割は、保護されたリソースのリストへのアクセスを許可する前に、認証要求が検証されることを保証することです。コントローラからAHに送信されるIH認証メッセージは、新しいIHが検証され、AHがこのIHに対して指定されたサービスのためのアクセスを許可すべきであることをAHに示します。

このメッセージはコントローラからAHに送信されますが、IHとコントローラの間の相互認証が確立した後に送信されることに注意してください。

0x05	JSON フォーマットのIH情報の配列

JSONの仕様例を以下に示します。

フォーマット	例
{"IH Authenticators": "IH": <IH/Device Pair>, "session_id": <256ビットIHセッションID>, "hmac_seed": <IH用256ビットSPA HMACシード>, "hotp_seed": <IH用256ビットSPA HOTPシード>, "id": [<256ビットのサービスIDの配列>] }	{"IH Authenticators": "IH": "IH/DeviceID", "session_id": "4562", "hmac_seed": "###", "hotp_seed": "###", "id":["123445678","9012345"] }

コントローラからAHに送信されるこのペイロードの情報は、AHがIHからの将来の接続を検証するのに十分である必要があることに注意してください。上の例では、AHはIH固有のデバイスID、セッションID、SPAシードを必要とします。実装によって、このアプローチは異なるかもしれません。

h. ログアウト要求メッセージ

ログアウト要求メッセージは、IHがSDPセッションを終了することを示すために、IHからコントローラに送信されます。コントローラから応答メッセージは送信されず、TLSおよびTCPコネクションは、IHまたはコントローラのいずれかによって終了される必要があります。ただし、IHはオンボードのままなので、将来的に新しいセッションを確立することは可能です。

0x07	コマンド指定データなし

z. ユーザ定義メッセージ

このコマンド（0xff）は、IHとコントローラ間の非標準的なメッセージの送受信のために予約されています。図11では、このメッセージは表示されていません。

0xff	ユーザー指定

IH-AHプロトコル

Initiating Host to Accepting Host プロトコルは、ネットワークルーティングとパケット配送を利用します。実装の詳細は、トランスポートの種類（例：TCP guaranteed deliveryあるいはUDP fire and forget）に依存します。

IHが有効なSPAパケットで接続し相互認証通信が確立された後にのみ、AHを通してサービスへの接続が動的に確立されることに注意することが重要です。そして、AHはIHが要求されたサービスにアクセスする権限があることを確認します。それまでは、AHによってサービスは隠されたままになります。

IHからAHへのシーケンス図

IHとAH間のプロトコルのシーケンス図の例を図12に示します。

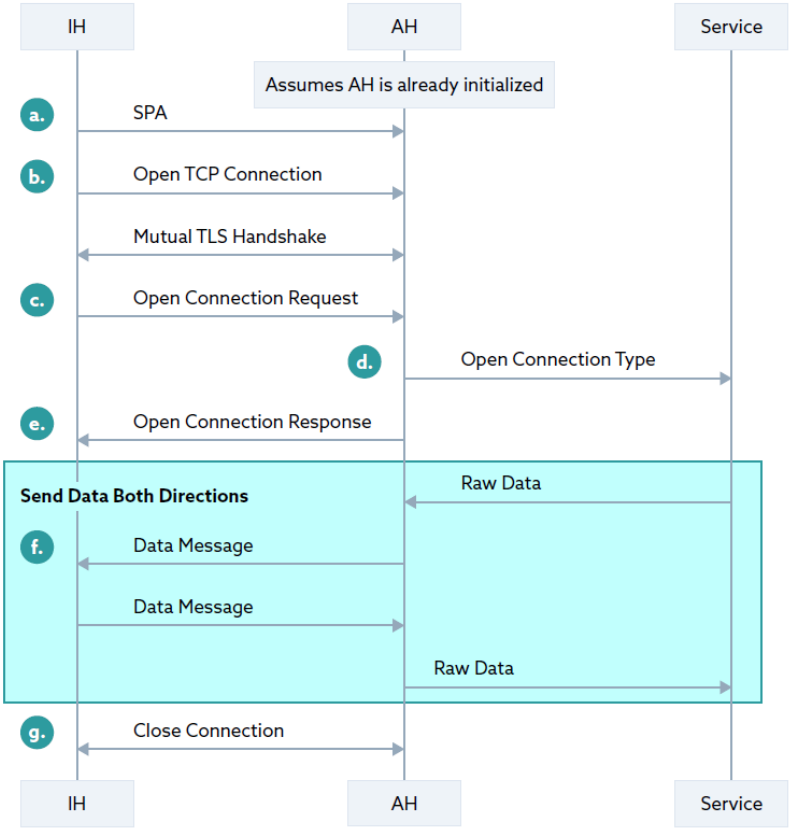


図12：IHがAHに接続し、その後サービスにデータを送信する

以下、IHとAHの間で受け渡される様々なメッセージとフォーマットを定義します。基本プロトコルは以下の形式になります：

コマンド（8ビット）	コマンド指定の長さでコマンド指定のデータ

a. SPA

SPAパケットは、IHからAHに送信され、前述したフォーマットに従って接続を要求します。

b. コネクションを開き、相互に認証された通信を確立

SPAパケットを送信後、IHはAHとのTCPコネクションを開こうとします。AHは、SPAパケットが有効であると判断した場合、このTCPコネクションの確立を許可します。この後、mTLS接続を確立するために必要な相互認証が行われます。

UDPベースのDTLSの場合、UDPはコネクションレス型プロトコルであるため、これは論理的な接続となります。

c. オープン接続リクエストメッセージ

IHはAHに対して、特定のサービスへの接続を開くことを示すオープンリクエストメッセージを送信します。

サービスIDは、リモートサービスごとにControllerが割り当てる一意の値です。Service IDは、IH-Controllerプロトコルの*IHサービス*メッセージの一部として、コントローラからIHに送信されているため、IHはService IDを認識します。

Session IDは、IHとAHによって、特定のリモートサービスの異なるTCP接続を区別するために使用されます。

0x07	Service ID - 256ビット識別子 Session ID - 256ビット識別子

d. 適切なコネクションタイプを開く

実装とSDP配備モデルに固有のこのステップは、AHがIHに代わってサービスへの接続を確立します。これは、短時間のHTTPS接続や、最初の交換の一部としてIHが提供するアプリケーションレベルの認証を必要とする接続（例：SSH）など、特定のプロトコルでは必要ないかもしれません。 SDP対応サービスは、アプリケーションプロトコルの外側で、AHからIHまたはユーザーのコンテキストを通信するために、この接続を確立する必要があるかもしれません。SDPの配備モデルによっては、この接続をネットワーク接続にしたり、ローカルホスト接続（たとえば、プロセス間通信）にしたりすることができます。

e. オープン接続レスポンスメッセージ

open-responseメッセージはAHからIHに送信され、オープン要求が成功したかどうかを示します。 Open Connection Request と Open Connection Response メッセージは、IHにとって非同期であるかもしれないので，サービスIDとセッションIDのコードはIHがこの戻り値を対応するリクエストと関連付けるために有用です.

0x08	Status Code (16ビット)	Service ID - 256ビット識別子 Session ID - 256ビット識別子

f. データメッセージ

データメッセージは、IHまたはAHのいずれかによって送信されます。オープンコネクションにデータをプッシュするために使用されます。レスポンスはありません。このメッセージは実装方法に依存します。SDPの実装によっては、このようなメッセージでアプリケーションデータをパッケージ化するものもあれば、別のアプローチを使用するものもあります。

0x09	Data Length (16ビット)	Service ID - 256ビット識別子 Session ID - 256ビット識別子

g. 接続終了メッセージ

g. 接続終了メッセージは、IHまたはAHのいずれかによって送信されます。これはAHによって接続が閉じられたことを示すか、IHが接続を閉じることを要求するために使われます。レスポンスはありません。

0x0A	Service ID - 256ビット識別子 Session ID - 256ビット識別子

z.ユーザー定義メッセージ

このコマンド（0xff）は、IH-AH間の非標準的なメッセージのために予約されています。このメッセージは図12には示されていません。

0xff	ユーザー指定

ロギング

サービスの可用性やパフォーマンス、サーバーのセキュリティ判断するためのログを作成することは、すべてのシステム、そしてゼロトラストの実現に必要な要件になります。

ログメッセージのフィールド

すべてのログは以下のフィールドを持ちます。

フィールド名	意味
time	ログレコードが生成された時刻
name	イベントに対する人間にとって読みやすい名前。注：ユーザー名、IPアドレス、ホスト名など、ミュータブルな data nuggets は含めないでください。その情報は、ログレコードの追加フィールドにすでに含まれています。
severity	このイベントの debug からcritical までの重要度（以下を参照）。
deviceAddress	ログレコードを生成したマシンのIPアドレス

オペレーションログ

以下は、ロギングを必要とする運用におけるユースケースとアクティビティの一覧です。

signature_id は、イベントの種類を識別するための識別子です。3カラム目には、特定のメッセージについて記録する必要がある追加フィールドが含まれています。

アクティビティ	signature_id	記録するデータ/情報
コンポーネントの起動、シャットダウン、再起動（例：コントローラ起動、ホスト再起動）	*ops:startup* *ops:shutdown* *ops:restart*	***reason*** *は、再起動やシャットダウンの理由を示します。* ***component*** *は、どのコンポーネントが影響を受けたかを示します。*
コンポーネント間接続（コントローラ、IH、AH、サードパーティコンポーネント、DB） アップ、ダウン、再接続	*ops:conn:up* *ops:conn:down* *ops:conn:reconnect*	***src*** は、報告エンティティから見た接続の発信元（IP アドレス） ***dst*** は、報告エンティティから見た IPアドレスとしての接続先 ***reconnect_count*** *は、再接続が何回試みられたか* ***reason*** は、通信がダウンした理由

セキュリティ/接続ログ

セキュリティログはSDPの中核であり、より大規模なインフラ攻撃を検知するための広い意味でも重要です。したがって、これらのログは、 Security Information and Event Management （SIEM）に転送されると非常に便利です。ゼロトラストの実装に必要なものになります。

signature_id は識別子であり、イベントの種類を識別することが可能です。3 カラム目には、特定の signature_id に対して記録される必要がある追加フィールドが含まれます。

アクティビティ	signature_id	記録するデータ/情報
AHログイン成功	*sec:login*	**src** コントローラから見たAHのIPアドレス **AH Session ID** AHのセッションID
AHログイン失敗	*sec:login_failure*	**src** コントローラから見たAHのIPアドレス **AH Session ID** AHのセッションID
IHログイン成功	*sec:login*	**src** コントローラから見たIHのIPアドレス **IH Session ID** IHのセッションID
IHログイン失敗	*sec:login_failure*	**src** コントローラから見たIHのIPアドレス **IH Session ID** IHのセッションID

SPA認証	*sec:auth*	IHからControllerへの**SPA**パケット AHからControllerへの**SPA**パケット IHからAHへの**SPA**パケット
コンポーネント認証（例えば、 IH -> Controller)	*sec:connection*	**IH Session ID** IHのセッションID **AH Session ID** AHのセッションID
受信拒否接続	*sec:fw:denied*	**src** 接続を試みた送信元 **dst** 接続を試行した送信先

ここでは、完全な停止状態を表す簡単なシナリオを示し、どのログエントリーがどこに書き込まれるかを示します。このシナリオでは、コントローラがダウンしたと仮定します。

コントローラがダウンする[ログが出ない、故障したコンポーネントはログが出せない]。

1. IHは Controller への再接続をn回試みます
   ops:conn:reconnect ログメッセージ
2. n 回後、クライアントは Controller への接続が切断されたことを宣言し、新しい Controller を探します。
   エラーの深刻度を付与した***ops:conn:down***ログメッセージ。
3. IHは新たに発見したControllerに接続します
   ops:conn:up ログメッセージ
4. 使用可能なコントローラが存在しない場合
   深刻度critical***のops:conn:down***ログメッセージ

また、同様のケースとして、警告なしにクライアントがダウンした場合（例：ノートパソコンが閉じられた）などが考えられます。この場合、AH と同様にControllerも接続の失敗を検出します。それぞれ、エラーの深刻度を示す ***ops:conn:down というログメッセージを記録します。***

以下は、完了したユーザーログイン（AHへのIH接続）の例になります：

5. IHはControllerに接続
   sec:auth ログメッセージ - ControllerへのSPA認証
   ops:conn:up ログメッセージ
6. IHはControllerと相互に認証します
   sec:接続ログメッセージ
7. IH が AH に接続
   sec:auth ログメッセージ
   ops:conn:up ログメッセージ

まとめ

SDPは、ゼロトラストの実装として有効です。[19] この改定された仕様により、企業はアプリケーション、ネットワーク、ユーザー、データを保護するためにゼロトラストのパラダイムを採用することが推奨されます。これは、クラウドへの移行や脅威の激化を考慮すると、非常に重要な課題となっています。

また、SDPは、企業のInfrastructure as a Serviceクラウドサービス、ネットワーク機能仮想化、Software-Defined Networking、IoTアプリケーションの利用を保護することが実証されています。

このSDP（Software-Defined Perimeter）仕様のバージョン2.0は、長い時間をかけて作られたものです。バージョン1.0は、約8年前の2014年4月に公開されました。この間、ゼロトラストの原則に対する熱意と採用が進み、それに伴ってSDPベースのソリューションに対する関心と導入が拡大してきました。

バージョン2.0仕様では、以下の領域を拡張、修正（明確化、拡張を通して）しています。

- SDPとゼロトラストの関係
- SDPのアーキテクチャと構成要素
- オンボーディングとアクセスワークフロー
- SPAメッセージのフォーマット、UDPの使用、およびその代替手段
- IoTデバイスとアクセスポリシーの初期検討

さらに、以下の3つのSDPサブプロトコルにおいて、シーケンス図を充実させ、コネクションやメッセージの説明を充実させました。

- AH to Controller
- IH to Controller
- IH to AH

SDP WGが今後の研究・出版に向けて検討しているテーマには、以下のようなものがあります:

- ゼロトラストポリシーモデルとSDP（すなわち、ポリシー決定点（PDP）と対応するポリシー実施点（PEP）を介したアクセス）
- SDPで実現するIoTのゼロトラスト
- SDPクレデンシャルの安全な保管とローテーション
- デバイスバリデーション

以上のテーマについて、今後のコラボレーションと情報公開を期待しています。

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.

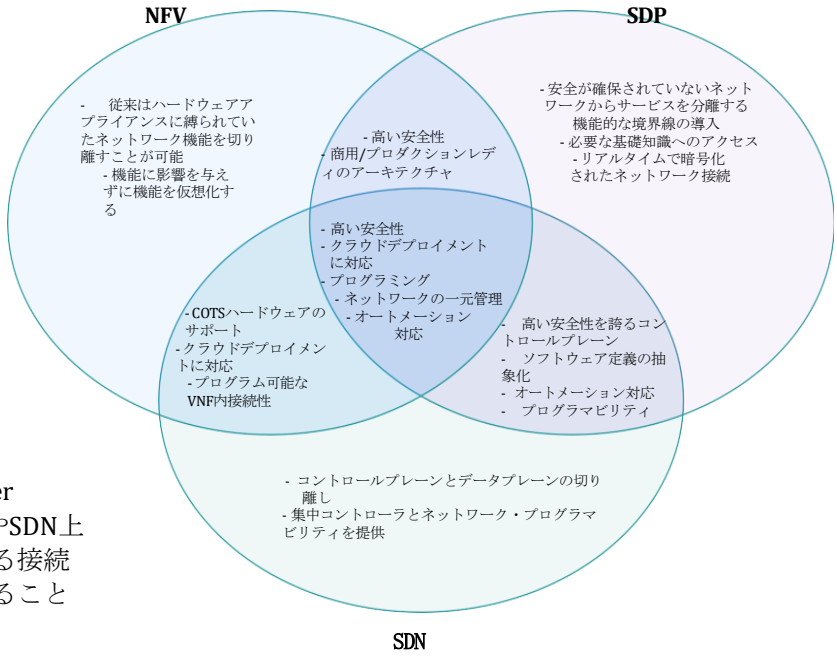
Ahmed Sallam, Ahmed Refaey, and Abdallah Shami, " On the Security of SDN: A Completed Secure and Scalable Framework Using the Software-Defined Perimeter," IEEE Access, Accepted, August 2019. (Impact factor: 4.098).

付録A：SDP、SDN、NFV

クラウドコンピューティング環境において、SDN（Software Defined Network）とNFV（Network Function Virtualization）の技術は、IH環境（フロントエンド）とAH環境（バックエンド）の両方の課題[20]に効率的に対応し、オンデマンド、従量制、サービスとしてのリソース提供といったメリットを提供するものです。SDNとNFVは、リソース（無線とコンピューティングリソース）をより良く利用するための異種ネットワークルーティングの採用とオーケストレーションに道を開くものです。IH環境の課題は、無線モバイルネットワークやエッジネットワークのアクセスレイヤーにおける通信のセキュリティに関するものであり、AH環境の課題は、ルーティング、スイッチング、セキュリティ、アカウンティング、課金など、ネットワークルーティングの機能に必要なネットワーク機能に関するものです。

NFVの大きな課題のひとつに、リソースの枯渇があります。特定の物理サーバーのリソースを集中的に使用するソフトウェアが、そのリソースを使い果たし、VMの可用性に影響を与える可能性があります。この状態は、物理サーバー内の共有環境が、特に複数のVMが同じリソース集約型ソフトウェアを同時に実行する場合に、リソース競合の深刻さを増大します。この問題は、SDP を使用することで解決できます。SDP コントローラが標準的な操作手順を定義し実装し、リソース枯渇によるスロットル（DoS と同様）を受けた VM を検出し、動的に救済策を講じることで、解決することができます。NVFに共通するもう1つのリスクは、セルフサービスポータルやクラウド管理コンソールを介したアカウントやサービスの乗っ取りです。ポータルやコンソールへのアクセスが、エンドユーザーに与えられた過剰な管理権限によってリスクへの露出を増加させます。この場合、SDPはこのリスクを排除し、ユーザーの役割と機能に基づいて選択的に管理制御を行う上で重要な役割を果たします。

クラウドサービスプロバイダは、ネットワーク管理を簡素化するためにSoftware-Defined Networks (SDN)をを採用しています。SDNの主な課題は、いかにして適切な認証、アクセス制御、データプライバシー、および、ネットワークルーティングのAPI駆動型オーケストレーションのためにデータの完全性を提供するかということです。



ここで、Software Defined Perimeter (SDP) は、ネットワークアクセスやSDN上のオブジェクト間の接続を制限する接続のオーケストレーションを提供することができます。

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

SDPとSDNが統合された結果、いくつかのメリットが生まれる可能性があります。特に、完全にスケーラブルで管理されたセキュリティソリューションが提供されます。

つまり、Software-Defined Network (SDN) と Network Function Virtualization (NFV) をネットワーク仮想化のトライアングルの2辺とし、Software-Defined Perimeter (SDP) を3辺として考えています。SDN と SDP の両方がネットワークの領域で動作し、名前が似ていても、SDP がゼロトラストの概念で大きな障害なしに信頼できるネットワーク接続を導入するのに対して、SDN はネットワークオペレーションを指揮する頭脳と考えることができます。

付録B：OSI/SDP コンポーネントマッピング

このセクションでは、OSIネットワーク層とクラウドレイヤー、クラウドスタック機能、およびSDPコンポーネントのマッピングを提供します。このマッピングの目的は、SDPが論理的にネットワークとクラウドモデルの異なるレイヤーでセキュリティを提供するのに役立つことを示すことです。

OSIレイヤ	クラウドレイヤー	クラウドスタック	SDPコンポーネント
アプリケーション	アプリケーション	エンドユーザーレイヤー ー アプリケーションとビジネス価値を提供します	SDPクライアント ー アプリケーションへのユーザーアクセスを提供します
プレゼンテーション	サービス	ミドルウェア ー アプリケーションがレイヤー内で使用する機能コンポーネント	SDPコントローラー ユーザーのリソースへのアクセスを仲介するためのミドルウェアとして機能します
セッション	イメージ	オペレーティングシステム ー 基盤となる仮想化を適切に管理します	SDPポリシー ー ファイアウォールルールの適用とセッション管理
トランスポート	ソフトウェア・デファインド・データセンター	Cloud API ー リソースプール/ユーザーに紐づく仮想化された資産の作成を可能にします	双方向TLS
ネットワーク	ハイパーバイザー	仮想化ー コンピューティング、ストレージ、モニタリングの仮想化を提供します	SDP Accepting Host (Gateway)
データリンク	インフラ	ハードウェア ー データセンター内の物理デバイス	n/a
物理	インフラ	ハードウェア ー データセンター内の物理デバイス	n/a