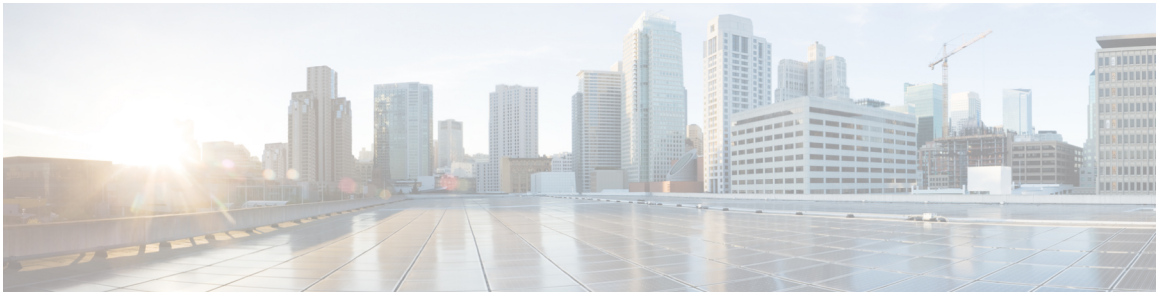
Trustworthy システムガイド

最終更新：2026 年 4 月 20 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

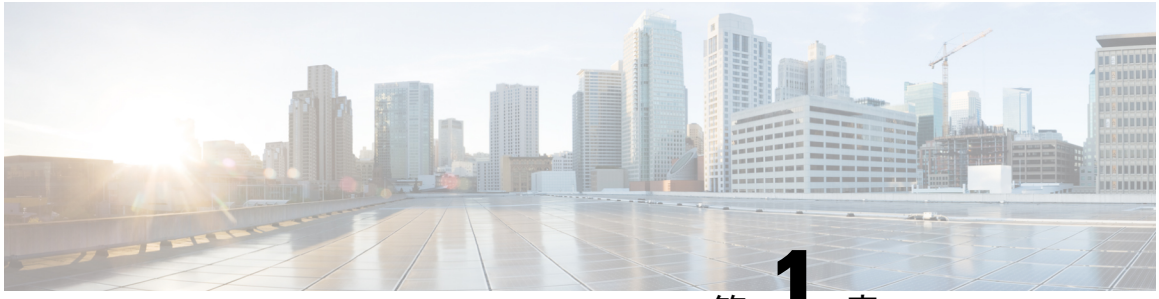
最初にお読みください



最初にお読みください

サポートされている機能のみが記載されています。プラットフォームでサポートされているすべての機能を確認または明確にするには、Cisco Feature Navigatorにアクセスします。

目 次



第 1 章

Trustworthy システム

Trustworthy システム

シスコの信頼できるテクノロジーにより、製品保証と基本的なセキュリティ機能が提供されます。これらの機能により、シスコのソリューションのセキュリティとレジリエンスが強化されます。

デバイスの偽造やハードウェアおよびソフトウェアへの悪意のある攻撃から保護し、デバイスの真正性と完全性を確認するために、シスコはデジタル署名付きソフトウェアイメージ、ハードウェアアンカー 型セキュアブート、Secure Unique Device Identifier（SUDI）、その他の信頼できるテクノロジーを利用しています。

これらのセキュリティ機能により、

- 偽造やソフトウェア変更からデバイスが保護され、
- 暗号化されたセキュアな通信を確立するのに役立ち、
- シスコのネットワークデバイスが意図されたように機能できるようになります。

信頼できるテクノロジーにより、デバイスのハードウェアおよびソフトウェアに対して自動化された完全性チェックが実行されます。デバイスへの信頼は、ハードウェアレベルから始まり、ブートプロセス、オペレーティングシステムのカーネル、およびオペレーティングシステムのランタイムまで継続されます。デバイスが侵害された場合は、ブートプロセスをシャットダウンしてシステムを保護することができます。

Trustworthy システムの必要性

複雑なコンピューティングネットワークと通信ネットワークを当てにして、サービス運用を中断なしで継続しています。ネットワークをセキュアな状態に保ち、ユーザーの信頼を維持するには、信頼性の高いデータと IT インフラストラクチャが必要です。いつでもどこからでも個人データにアクセスできるようになるため、すべてのネットワークにわたって一貫性のあるアクセス権とセキュリティが期待されます。

攻撃者はますます攻撃的になっているため、急速に変化する脅威状況にさらされています。悪意のある攻撃者による攻撃や、偽造または改ざんされた製品からネットワークを保護する必要があります。Trustworthy システムは、ネットワークセキュリティを維持し、ユーザーの信頼を保護し、ますます巧妙になっている脅威に備えることに役立ちます。

SUDI

SUDI は、X.509 証明書を使用してデバイスに改ざん防止 ID を提供します。秘密キーは製造時にシスコのトラストアンカーモジュール（TAm）に安全に格納されるため、デバイスの ID が確認されて、偽造から保護されるようになります。

デバイスは、製品 ID とシリアル番号を組み合わせた SUDI によって一意に識別されます。X.509 バージョン 3 証明書は、IEEE 802.1AR に準拠し、RSA または ECC 暗号化が使用されているため、ID の複製やスプーフィングが防止され、偽造のリスクから機器を保護できます。

トラストアンカーモジュール

Cisco TAm は、シスコデバイスに埋め込まれる改ざん防止チップです。これには、不揮発性セキュアストレージが含まれており、デバイスの信頼チェーンの基盤として機能します。メーカーの公開キーは、デバイスの TAm チップ内に安全に格納されます。

デバイスには、ACT および ACT2 と呼ばれる専用ハードウェアチップをデバイス内に埋め込むことによる偽造防止テクノロジー（ACT）が含まれています。

TAm には次の機能があります。

ID

製造時に、デバイスの ACT2 チップは、X.509v3 ECDSA または RSA 証明書（またはその両方）としての Cisco SUDI、およびキーペアと証明書チェーンを受信します。SUDI は、ハードウェア偽造防止チェックの基礎を形成し、デバイスの初期ネットワーク ID を確立するのに役立ちます。

エントロピー

ACT2 は、NIST SP 800-90B 準拠のエントロピーソースを備えているため、ホストベースの疑似乱数ジェネレータのシードに最適です。

キー管理

ACT2 は、対称キーだけでなく、ECC および RSA の非対称キーペアを生成できます。重要なのは、対称キーおよびこれらのキーペアの秘密部分がチップから解放されないことです。これらの保護されたキーへのアクセスは、暗号化 API を介してのみ行われ、ACT2 によって生成されたキーペアに対して証明書を登録できます。

セキュアなストレージ

ACT2 は、物理的に改ざんされる恐れがない約 50 KB のホストデータストレージを提供します。そのため、ライセンスなどの機密情報やクレデンシャルなどの機密データを格納するのに最適なリポジトリです。重要なのは、重要なデバイスキーやパスワードもこの安全なストレージ内で保護されることです。

デバイス内の TAm および SUDI は、モバイルデバイスの IMEI 番号と同様に、一意のハードウェア識別子として機能し、追加機能も提供します。

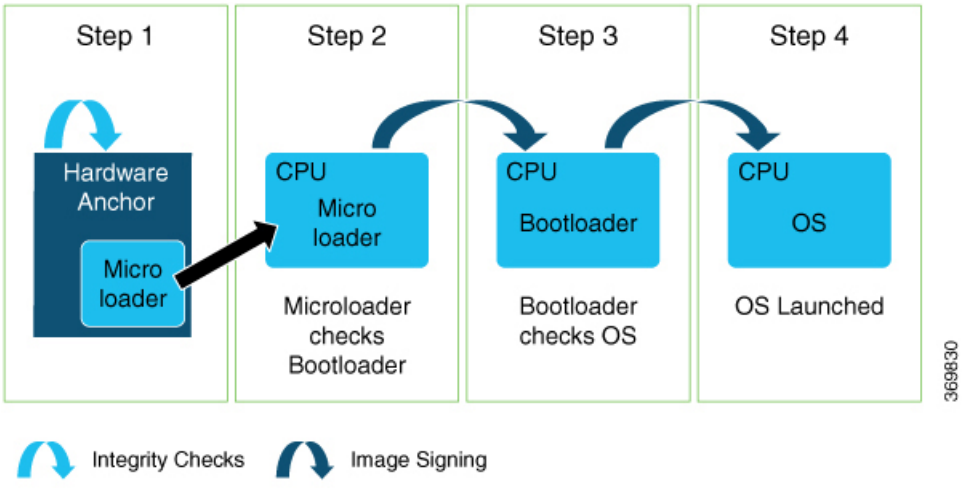
TAm は、アプリケーションが TAm ライブラリを介してアクセスする暗号キー用のセキュアな不揮発性ストレージを備えています。デバイスが起動すると、SUDI と TAm に格納されている起動測定値によって、デバイスが正当であることが確認され、ブートプロセスが期待どおりに動作すること、および改ざんの検出に役立ちます。

セキュアブート

Cisco Secure Boot により、デバイスで実行されるコードの真正性と完全性が保証されます。ハードウェアアンカー型セキュアブートにより、改ざん防止ハードウェア内のマイクロローダー（ブートコード最初の部分）が保護されます。これにより、信頼の基点が作り出され、汚染されたネットワークソフトウェアがシスコ デバイスで実行されるのを効果的に防止できます。

シスコは、製造元の秘密キーを使用してすべてのイメージにデジタル署名します。これは、セキュアな監査済み環境で行われます。キーペアは、2048 ビットモジュラスの RSA 暗号化を使用して生成されます。

デバイスは、ブート後に SUDI を介して TAm で認証されるため、ハードウェアが偽造または侵害されたユニットではなく、本当に純正シスコ製品であることが保証されます。



ハードウェアに直接埋め込まれている公開キーには、TAm ライブラリを介してアクセスできます。Cisco IOS XE ソフトウェアイメージは、ビルド DevOps リリース環境から削除されることがない秘密キーを使用して署名されます。ブートプロセスで、これらのキーが比較され、ハードウェアとソフトウェアの両方がシスコによって署名されていることが確認されます。

TAm が CPU にロードされる前にマイクロローダーの完全性を確認するときにセキュアチェーンが開始されます。次に、マイクロローダーが TAm の API を使用して BIOS またはブートローダーをチェックし、ロードする前にイメージが正当であることを確認します。この検証は、Cisco IOS XE イメージのロード時にも行われます。

CPU がイメージをロードすると、オペレーティングシステムは TAm API を使用して、KLM、RPM、ASIC SDK などのソフトウェアモジュールを確認してからアクティブ化します。

チップガード（チップ保護）

シスコ デバイスには、ハードウェアの完全性を保証するチップ保護機能が組み込まれています。この保護メカニズムでは、製造時に各デバイスの署名が記録され、デバイスのブートごとに署名が比較されるため、デバイスの周辺機器が偽造されていないことが保証されます。

製造時間データベース

製造時間データベースは、シスコの ASIC、CPU、SoC、およびその他のデバイスの一意の ID と、ボードに特有のデバイスタイプのオリジナルコピーです。ほとんどの場合、一意の ID は、デバイスのシリアル番号またはそのデバイスのその他の適切な値です。製造データベースは、ボードに特有の Known Good Values（KGV）データベースであり、製造プロセスの一環として TAm デバイスにプログラミングされます。

収集対象データベース

収集対象データベースは、ボードがブートされて TAm デバイスに拡張されるたびにファームウェアによって収集されます。測定値は、ファームウェアまたはシステムドライバによって収集されます。

BIOS ブートプロセスでは、TAm ライブラリが組み込まれて、収集対象データベースに値が入力されます。BIOS は初期化の一環としてさまざまなハードウェアコンポーネントを検出し、検出されたデバイスが製造時間データベースの一部である場合は、TAm ライブラリ API を使用してデバイスタイプと一意の ID を記録します。すべてのデバイスタイプと一意の ID が収集対象データベースに書き込まれると、プラットフォームのオペレーティングシステムが TAm ライブラリ API を呼び出して、製造時間データベースと照合して収集対象データベースを検証します。不一致がある場合、プラットフォームはブートプロセスを保留します。

乱数生成とエントロピーソース

強力な乱数生成（RNG）は暗号化の中核を成しており、弱い RNG では暗号化システム全体が弱体化する可能性があります。乱数ジェネレータは、暗号キーの作成、ユーザーと Web サイト間の非常にセキュアな通信の確立、電子メールアカウントのパスワードのリセットにおいて、重要な役割を果たします。確実にランダムでなければ、攻撃者はシステムが生成しようとするものを予測し、アルゴリズムを弱体化させることができます。シスコのデバイスでは Linux の RNG が使用されています。RNG では通常、ハードウェア乱数ジェネレータ（HRNG）から

得られるランダムな値がシードとして使用されるため、推測することは不可能です。ハードウェアには、NIST 仕様に準拠していて、トラストアンカー内の真のランダムソースからエントロピーを抽出するはるかに効果的な RNG を提供できる、トラストアンカーモジュールも組み込まれています。

マルチステージ BIOS

BIOS は、RAM 内で完全にロード、検証、および実行して BIOS を外部変更（Time-of-Check to Time-of-Use 攻撃など）から保護できるように、複数の小さな部分に分割されています。[1]を押します）。BIOS は次のもので構成されています。

- Pre-EFI 初期化（PEI）
- ファームウェア依存関係モジュール（FDM）
- ドライバ実行環境（DXE）

デバイスにマルチステージ BIOS があると、Cisco BIOS をバイパスするのが非常に困難になります。BIOS に介入すると、ブートローダーによる OS イメージのロードが停止されます。

ランタイム防御（RTD）

ランタイム防御は、実行中のソフトウェアに対する悪意のあるコードのインジェクション攻撃を対象としています。シスコのランタイム防御には、アドレス空間配置のランダム化（ASLR）、組み込みのオブジェクトサイズチェック（BOSC）、X-Space などがあります。これらの防御策により、攻撃者が実行中のソフトウェアの脆弱性を悪用することが困難または不可能になります。

アドレス空間配置のランダム化（ASLR）

アドレス空間配置のランダム化（ASLR）は、シスコのデバイスのすべてのプロセスとカーネルのセクションの場所をランダム化して、攻撃者が既存の脆弱性を悪用するのを困難にする、重要なセキュリティ強化機能です。ASLR は、実行保護と併用される保護です。実行保護では、許可されていない領域からのコードの不用意な実行が阻止され、実行可能領域へのコードの上書きが禁止されます。

プロセスに対する ASLR 機能は、シスコのバイナリとサードパーティのバイナリに分類され、どちらも ASLR をサポートする必要があります。ASLR をサポートするには、シスコおよびサードパーティのバイナリと共有ライブラリが、正しいフラグを使用してビルドされている必要があります。シスコのバイナリおよびサードパーティの共有オブジェクトでは、シスコのバイナリ自体のランダム化を侵害しないようにライブラリがランダム化されるようにしておく必要があります。サードパーティのバイナリおよび共有ライブラリでは、それらをランダム化するためにベンダーのサポートが必要になる場合があります。

Linux カーネルに対する ASLR 機能では、ブート時にカーネルコードが配置される場所をランダム化することで、Linux カーネルイメージを実行するためのアドレス空間のランダム化がサ

[1] Time-of-Check to Time-of-Use（TOC/TOU）攻撃は、プログラムが条件をチェックし、そのチェック結果に基づいてアクションを実行するときに発生する競合状態の脆弱性の一種です。ただし、別のプログラムによってチェック時から使用時までに条件が変更された場合に、変更された条件に基づいてアクションが実行されるため、意図しない結果になる可能性があります。

ポートされます。カーネルASLRサポートはシスコのデバイスに存在し、ハッカーが悪意のあるコードを挿入するのを阻止します。

実行保護（X-Space）

実行保護（X-Space）は、シスコのデバイスで最も重要なセキュリティ保護の 1 つです。この機能により、デバイスに対する実行保護が有効になります。これにより、許可されていない領域からのコードの実行が阻止され、実行可能領域へのコードの上書きが禁止されます。

X-Space は、デバイスへのハッカーの侵入を阻止し、デバイスをセキュアな状態に保ちます。

オブジェクトサイズチェック（OSC）

バッファオーバーフローはおそらく、ソフトウェアのセキュリティ脆弱性の最もよく知られた形式です。バッファオーバーフロー状態は、バッファに保持できるよりも多くのデータをプログラムがバッファに配置しようとした場合、またはバッファ領域を超えてメモリ領域にプログラムがデータを配置しようとした場合に発生します。この場合のバッファは、文字列から整数の配列までのあらゆるもの入れるために割り当てられた、メモリの連続セクションです。割り当てられたメモリのブロック境界の範囲外に書き込むと、データの破損、プログラムのクラッシュ、または悪意のあるコードの実行が発生する可能性があります。シスコのデバイスは、書き込みコールの前にオブジェクトサイズチェックを行うことによって、CまたはC++コードでのバッファオーバーフローを判別するための完全な保護を備えています。

SafeC ライブラリ

Cisco IOS XE ソフトウェアでは、より安全でよりセキュアな C または C++ 言語プログラミングを促進し、ISO/IEC 9989:2011（C11）仕様に基づいている、効率的なライブラリ関数が使用されています。いくつかの標準Cライブラリ関数は、より巧妙な攻撃の開始点として機能する脆弱性の影響を受けやすい状態になっています。

SafeCは、一貫性のある命名スキーマで標準関数の「安全な」代替を提供する一方で、バッファオーバーフローによるセキュリティ侵害を軽減することを目的としており、ネイティブライブラリに存在しない可能性のある境界チェックを提供し、文字列の終端および切り捨てのエラーを防止します。この SafeC により、シスコのデバイスのハードウェアがバッファ オーバーフロー攻撃から保護されます。

シスコの署名付きカーネルモジュール

署名付きカーネルモジュールにより、デバイスにロードされるすべてのカーネルモジュールが真性であり、変更されていないことが保証されます。これにより、未承認または信頼できない実行可能コードが従来の方法でカーネルにロードされることが阻止されます。

シスコによって署名されていないモジュールは、Cisco IOS XE ソフトウェアでは使用できません。この機能により、未承認のソフトウェアがシスコのデバイス上で実行されることが阻止されます。これらの強化機能により、Cisco IOS XE ソフトウェアとそのコンポーネントが攻撃から保護されます。

セキュア JTAG（sJTAG）

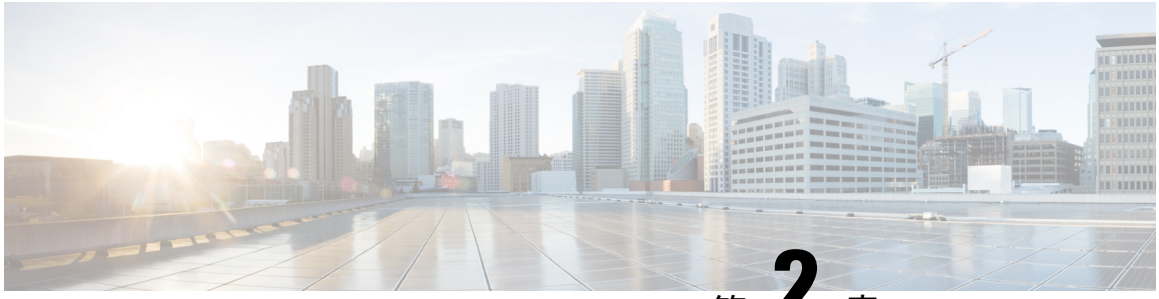
JTAG は、FPGA のプログラミングにも採用されており、CPU デバッグアクセスポートを提供します。多くの場合、組み込み CPU にアクセスしてファームウェアイメージの取得、メモリのダンプ、およびソフトウェア実行のモニタリングを可能にするために必要なものは、ラップトップと JTAG デバッガだけで済みます。高度なツールセットと組み合わされた小規模なインターフェイスにより、システムを悪用するためのポータブルでありながら強力な手段が攻撃者に提供されます。

シスコのデバイスにセキュア JTAG を備えることにより、知的財産（IP）の盗難を軽減し、メモリからパスワードやキーの盗むことを阻止することができます。

安全消去

セキュア消去機能により、シスコのデバイス内のすべてのお客様情報が消去されます。セキュア消去は、Return Merchandise Authorization（RMA）、アップグレードや交換、システムのサポート終了によって製品を除去するために、識別可能な顧客情報をすべて削除する操作です。

- デバイスの返品許可（RMA）：RMA のためにデバイスをシスコに返却する必要がある場合は、そのデバイスの RMA 証明書を取得する前に、お客様特有のデータをすべて削除します。
- 侵害を受けたデバイスのリカバリ：デバイスに保存されているキーマテリアルまたはクレデンシャルが侵害を受けた場合は、デバイスを初期設定にリセットし、デバイスを再設定してください。



第 2 章

BIOS 保護

BIOS 保護

BIOS 保護機能によって、ゴールデン ROMMON イメージに対する書き込み保護が提供され、そのイメージへのアップグレードが安全に行われるようになります。

BIOS 保護機能がないと、ソフトウェアアップグレード時に悪意のあるコードによってゴールデン ROMMON が破損する可能性があります。

プライマリ ROMMON とゴールデン ROMMON

ROMMON イメージは、プライマリ ROMMON およびゴールデン ROMMON の両方として SPI フラッシュデバイスに格納されます。

プライマリ ROMMON は、デバイスの電源がオンになるか再起動されるたびに、デバイスをブートするために使用されます。プライマリ ROMMON が破損した場合、デバイスはゴールデン ROMMON を使用して IOS XE ソフトウェアイメージをブートします。

デバイスがプライマリ ROMMON からブートした場合、ゴールデン ROMMON はロックされたままです。

BIOS 保護機能が有効になっていると、ゴールデン ROMMON は書き込み保護されるため、フラッシュユーティリティの標準アップグレードメカニズムを使用してアップグレードすることはできません。ゴールデン ROMMON のアクセスポリシーは、FPGA ファームウェアによって適用され、ゴールデン ROMMON SPI フラッシュデバイスでの書き込みおよび消去コマンドなどの不正な操作がブロックされます。

（注） ゴールデン ROMMON のアップグレードは、セキュアブート FPGA がアップグレードされた後にのみ利用可能になります。

スタンドアロンデバイス、高可用性デバイス、および SVL デバイスでのアップグレード

アップグレードプロセスは、スタンドアロン、高可用性、および SVL デバイスごとに異なります。アップグレードプロセスの仕組みを次の表に説明します。

表 *1:* スタンドアロンデバイス、高可用性デバイス、および *SVL* デバイスでのアップグレード

デバイス設定	アップグレード プロセス
スタンドアロンデバイス	デバイスをインストールモードでアップグレードすると、デバイスのブート時にプライマリ ROMMON が自動的にアップグレードされます。ゴールデン ROMMON をアップグレードするには、カプセルアップグレードプロセスを使用します。
高可用性デバイスおよび StackWise Virtual デバイス	高可用性セットアップでは、デバイスに対して In-Service Software Upgrade（ISSU）を行います。この ISSU プロセスには FPGA のアップグレードが含まれます。 リロードを使用してインストールモードでアップグレードする場合は、一度に 1 つのスーパバイザをリロードします。スタンバイスーパバイザを ROMMON 状態にして、アクティブなスーパバイザを最初にブートします。アクティブなスーパバイザで ROMMON のアップグレードが完了したら、FPGA およびソフトウェアイメージもアップグレードします。

カプセルアップグレード

プライマリ ROMMON、プライマリ FPGA、およびゴールデン FPGA（セキュアブート FPGA）は、デバイスのブート時に自動的にアップグレードされます。対照的に、ゴールデン ROMMON をアップグレードできるのはカプセルアップグレード プロセスを使用する場合のみであるため、重要なブートコンポーネントに対するセキュリティが強化されます。

カプセルアップグレードでは、セキュアなアップデートカプセルが作成されてデジタル署名され、プライマリ ROMMON によってそのカプセルが使用されて、認証後にゴールデン ROMMON がアップグレードされます。このプロセスにはセキュアなフラッシュ証明書が必要であり、それはプロダクトキーおよびアップデートカプセルの真正性を確認するためにプライマリ ROMMON イメージに含まれている ID を使用して生成されます。カプセル自体は、セキュア

なフラッシュ証明書とセキュアブート 16 MB フラッシュイメージを使用して作成され、アップグレードの完全性と真正性を保証するために署名されます。

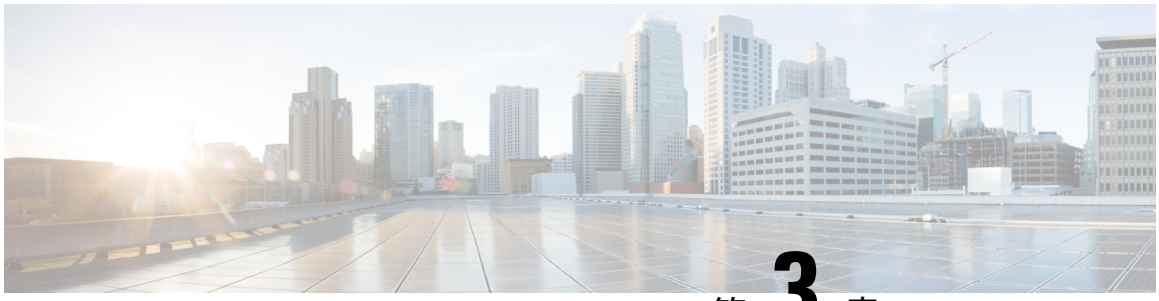
デバイスがブートすると、プライマリ ROMMON によってゴールデン ROMMON のカプセルアップグレードが開始されます。ゴールデン ROMMON のカプセルアップグレードを手動で行うには、スイッチで **upgrade rom-monitor capsule golden** コマンドを使用するか、スイッチスタックで **upgrade rom-monitor capsule golden switch** コマンドを特権 EXEC モードで使用します。

カプセルアップグレードの仕組み

process_workflow

ここでは、カプセルアップグレード時の動作について説明します。

1. デバイスは、セキュアブート FPGA アップグレードが有効になっているかどうかを確認します。セキュアブート FPGA アップグレードが有効になっていない場合は、アップグレードが停止します。
2. デバイスは、ブートローダー保護が有効になっているかどうかを確認します。ブートローダー保護が有効になっていない場合は、プライマリ ROMMON、ゴールデン ROMMON、およびプライマリ FPGA のワンタイムアップグレードが開始されます。
3. ブートローダー保護がすでにアクティブになっている場合は、Cisco IOS XE がセキュアなアップデートカプセルをブートフラッシュにコピーし、デバイスを再起動します。
4. デバイスが再起動すると、セキュアなアップデートカプセルが選択されて、アップグレードが実行されます。



第 3 章

ブート整合性の可視性

ブート整合性の可視性

ブート整合性の可視性は、ROMMON ソフトウェアを検証して完全性を確保することにより、ハードウェア トラストアンカーとして機能します。

ブート整合性の可視性によって、シスコのプラットフォームのプラットフォーム ID とソフトウェアの整合性情報の両方が可視化され、実用可能になります。プラットフォーム ID とは、製造時に割り当てられた一意の ID であり、これによって各デバイスを確実に識別できます。ソフトウェア整合性には、ブート整合性測定値のキャプチャを伴います。これは、プラットフォームが、信頼できるコードを使用してブートしているかどうかを判断するのに役立ちます。

お使いのシスコデバイスでは、ブート整合性の可視性機能がサポートされています。

ブート整合性の可視性の仕組み

process_summary

Cisco IOS XE ソフトウェアイメージをシスコデバイスにコピーすると、ROMMON ブート ROM がシスコのリリースキーを使用してイメージを検証します。このリリースキーは、シスコのビルドサーバーに安全に保存されているリリース秘密キーに対応する公開キーです。公開リリースキーは ROMMON 内に埋め込まれているため、デバイスはブートに Cisco IOS XE ソフトウェアイメージの真正性と完全性を検証できます。

process_workflow

ROMMON は、デバイスをブートする際に、次の手順に従って署名付き Cisco IOS XE ソフトウェアイメージを検証します。

1. Cisco IOS XE ソフトウェアイメージを CPU メモリにロードします。
2. Cisco IOS XE ソフトウェアパッケージのヘッダーを調べます。
3. イメージに対して非セキュア整合性チェックを実行して、ディスクまたは TFTP でファイル破損が生じていないことを確認します。このチェックでは、非セキュア SHA-1 ハッシュが使用されます。

（注） この手順では、ディスク、ファイル転送、またはコピーのエラーによる不注意の破損がチェックされます。これはイメージコード署名の一部ではないため、意図的なイメージ改ざん性は検出されません。

4. シスコの RSA 2048 ビット公開リリースキーを ROMMON ストレージからコピーし、シスコの RSA 2048 ビット公開リリースキーが改ざんされていないことを検証します。
5. パッケージのヘッダーからコード署名用署名（SHA-512 ハッシュ）を抽出し、シスコの RSA 公開リリースキーを使用して検証します。
6. Cisco IOS XE ソフトウェアパッケージの SHA-512 ハッシュを計算してコード署名の検証を行い、コード署名用署名と比較します。これで署名付きパッケージの検証が実行されたことになります。
7. Cisco IOS XE ソフトウェアパッケージのヘッダーを調べて、プラットフォームタイプと CPU アーキテクチャの適合性を検証します。
8. Cisco IOS XE パッケージから Cisco IOS XE ソフトウェアを抽出してソフトウェアをブートします。

Result

（注） イメージコード署名の検証は、ステップ 4、5、および 6 で行われます。これは、2048 ビット RSA キーで暗号化された SHA-512 ハッシュを使用した、イメージのセキュアコード署名チェックです。このチェックは、意図的なイメージの改ざんを検出することを目的としています。

このソフトウェアがシスコのビルドサーバーによって生成されたものではない場合は、署名の検証が失敗します。デバイスの ROMMON がイメージを拒否してブートを停止します。

署名の検証に成功すると、デバイスはイメージをブートして Cisco IOS XE ランタイム環境に入ります。

イメージ署名

シスコのビルドサーバーが Cisco IOS XE ソフトウェアイメージを生成します。Cisco IOS XE ソフトウェアイメージは、ビルド時にデジタル署名されます。バイナリイメージファイル全体に対して SHA-512 ハッシュが生成され、このハッシュがシスコの RSA 2048 ビット秘密キーで暗号化されます。

ソフトウェアイメージとハードウェアの確認

ここでは、スイッチのブート時に作成されたチェックサムレコードを取得する方法について説明します。特権 EXEC モードで次のコマンドを入力します。

表 *2:* チェックサムレコードを取得するコマンド

コマンド	説明
show platform sudi certificate [**sign** [**nonce** *nonce*]]	特定の SUDI のチェックサム レコードを表示します。 • （任意）**sign**： 署名を表示します。 • （任意）**nonce**： ナンス値を入力します。
show platform integrity [**sign** [**nonce** *nonce*]]	ブート段階のチェックサム レコードを表示します。 • （任意）**sign**： 署名を表示します。 • （任意）**nonce**： ナンス値を入力します。

プラットフォーム ID の確認

次に、PEM形式でセキュアな固有デバイス識別子（SUDI）チェーンを表示する例を示します。

SUDIにエンコードされるのは、個々のデバイスの製品IDとシリアル番号であり、何千ものデバイスからなるネットワーク上でデバイスを一意に識別できます。

- 最初の証明書は Cisco Root CA 2048 であり、
- 2 つ目はシスコの下位 CA（ACT2 SUDI CA）です。

  どちらの証明書も、https://www.cisco.com/security/pki/ で公開されているものと一致しているかを確認できます。
- 3 番目は SUDI 証明書です。

（注） 次のコマンドを実行すると、CLI に「**% Please Try After Few Seconds**」というメッセージが表示されることがあります。これはCLIの障害を示すものではありませんが、必要な出力を取得するために必要な基盤となるインフラストラクチャの設定を示します。数分間待ってから、コマンドをもう一度試します。

```
Device# show platform sudi certificate sign nonce 123
-----BEGIN CERTIFICATE-----
MIIDITCCAgmgAwIBAgIJAZozWHjOFsHBMA0GCSqGSIb3DQEBCwUAMC0xDjAMBgNV
BAoTBUNpc2NvMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwOTkwIBcNMTYwODA5
MjA1ODI4WhgPMjA5OTA4MDkyMDU4MjhaMC0xDjAMBgNVBAoTBUNpc2NvMRswGQYD
VQQDExJDaXNjbyBSb290IENBIDIwOTkwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAw
ggEKAoIBAQDTtuM1fg0+9Gflik4axlCK1I2fb3ESCL8+tk8kOXlhfrJ/zlfRbe60
xRP0iUGMKWKBj0IvvWFf4AW/nyzCR8ujTt4a11Eb55SAKXbXYQ7L4YMg+lmZmg/I
v3GJEc3HCYU0BsY8g9LuLMvqwiNmAwM2jWzNq0EPArt/F6RiQKq6Ta3e7VIfDZ7J
65OA2xASA2FrSe9Vj97KpQReDcm6G7cqFH5f+CrdQ4qwAa4zWNyM3kOpUb637DNd
9m+n6WECyc/IUD+2e+yp21kBZIKH7JvDpu2U7NBPfr52mFX8AfCZgkXV69bp+iYf
saH1DvXIfPpNp93zGKUSXxEj4w881t2zAgMBAAGjQjBAMA4GA1UdDwEB/wQEAwIB
BjAPBgNVHRMBAf8EBTADAQH/MB0GA1UdDgQWBBQ4lVcPNCNO86EmILoUkcdBiB2j
WzANBgkqhkiG9w0BAQsFAAOCAQEAjeKZo+4xd05TFtq99nKnWA0J+DmydBOnPMwY
lDrKfBKe2wVu5AJMvRjgJIoY/CHVPaCOWH58UTqfji95eUaryQ/s36RKrBgMMlwr
WNItxE625PHuaN6EjD1WdWiRMZ2hy8F4FCKz5hgUEvN+PUNZwsPnpU6q3Ay0+11T
4TriwCV8kJx3cWu0NvTypYCCXMscSfLFQR13bo+1z6XNm30SecmrxkmQBVMqjCZM
VvAxhxW1iGnYdPRQuNqt0xITzCSERqg3QVVqYnFJUkNVN6j0dmmMVKZh17HgqLnF
PKkmBlNQ9hQcNM3CSzVvEAK0CCEo/NJ/xzZ6WX1/f8Df1eXbFg==
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIEZzCCA0+gAwIBAgIJCmR1UkzYYXxiMA0GCSqGSIb3DQEBCwUAMC0xDjAMBgNV
BAoTBUNpc2NvMRswGQYDVQQDExJDaXNjbyBSb290IENBIDIwOTkwIBcNMTYwODEx
MjAyODA4WhgPMjA5OTA4MDkyMDU4MjdaMDExHzAdBgNVBAMTFkhpZ2ggQXNzdXJh
bmNlIFNVREkgQ0ExDjAMBgNVBAoTBUNpc2NvMIIBIjANBgkqhkiG9w0BAQEFAAOC
AQ8AMIIBCgKCAQEAvdzeSWdDI6lRZDYRvA6JqaRvQyy6Dx1WaqI82UeKR4ZRn0ef
xMGvp4c88/VMS8WSjQO1qolMfMxqHkcSiFBOULx6Trquw4TrEf9sIuzvgJvDaEa8
IllXPwtPtNqZEIWi8jlinz2uGam93KuGPcioHfruzbDKWHL/HWFGYMgz+OKwhD3J
4NRySknQvUovfV8eWLeVOqW8rbnG3TZxv5VexOiK4jL3ObvsQPuAWUwUoo7nuFlE
GTG/VCeyCe/H8+afIScbZOkI9xejtckflnBYFVCyFxzm2H3YZatb6ohbyRXLtOPj
T3SJ+OOoYMlSLd28z727LpRbFFLGYhyWxEXDuQIDAQABo4IBgjCCAX4wDgYDVR0P
AQH/BAQDAgEGMBIGA1UdEwEB/wQIMAYBAf8CAQAwfwYIKwYBBQUHAQEEczBxMEEG
CCsGAQUFBzAChjVodHRwczovL3d3dy5jaXNjby5jb20vc2VjdXJpdHkvcGtpL2Nl
cnRzL2NyY2EyMDk5LmNlcjAsBggrBgEFBQcwAYYgaHR0cDovL3BraWN2cy5jaXNj
by5jb20vcGtpL29jc3AwHwYDVR0jBBgwFoAUOJVXDzQjTvOhJiC6FJHHQYgdo1sw
UgYDVR0gBEswSTBHBgorBgEEAQkVAR4AMDkwNwYIKwYBBQUHAgEWK2h0dHA6Ly93
d3cuY2lzY28uY29tL3NlY3VyaXR5L3BraS9wb2xpY2llcy8wQwYDVR0fBDwwOjA4
oDagNIYyaHR0cDovL3d3dy5jaXNjby5jb20vc2VjdXJpdHkvcGtpL2NybC9jcmNh
MjA5OS5jcmwwHQYDVR0OBBYEFOpro7nBE5d+G/s6jWhgBzlfh0j6MA0GCSqGSIb3
DQEBCwUAA4IBAQBcqYEOgAHhGWKndwM901XX2Enh4hjXR5avDg7G/f6Tb9H509dt
QW+AeZGEghhwUrw1EeG79tHkncAe+m+64xMC1ttyI1RSyn8rBqQYkXnnCRbtF/Nw
pQe5fjvdeIFWJhUI16TOt/ZlkNnWnLsUU1alZmN+J/FhSr8VTJWGRM9gY8hefH8f
5U7LMiDXxsFVHB7R6KGNjvtawrl6W6RKp2dceGxEIIvMahgMWWHHiWOQAOtVrHuE
NEjYR/7klLLwdgQF/NNCA2z47pSfMFnBcr8779GqVIbBTpOP2E6+1pBrE2jBNNoc
uBG1fgvh1qtJUdBbTziAKNoCo4sted6PW2/U
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIFQzCCBCugAwIBAgIKCFlwlokGmWc/RzANBgkqhkiG9w0BAQsFADAxMR8wHQYD
VQQDExZIaWdoIEFzc3VyYW5jZSBTVURJIENBMQ4wDAYDVQQKEwVDaXNjbzAgFw0y
NTA1MDgxMTM1NDJaGA8yMDk5MDgwOTIwNTgyNlowazEmMCQGA1UEBRMdUElEOkM5
MzUwLTQ4VFggU046RlZIMjkxOUw4TVIxDjAMBgNVBAoTBUNpc2NvMRgwFgYDVQQL
Ew9BQ1QtMiBMaXRlIFNVREkxFzAVBgNVBAMTDlE1Q0ItRTVYOS1QRkpRMIIBIjAN
BgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA10Xb/EGY8VGXQcgLCgML0yMoR8bx
9cqgHYZ594RkxUt+qO3xvvxgXVPZmK4t+6h4z5/up0hXkZD1UYlXDeOVmQ8Z5rFV
```

```
gRAAr1a8LUWo5pSympQ8kP6Dl3WT9RM9Mu42S7O41W91e59cyTszRMoTUZGqQV9X
UkL8TWLd4ZnM56ZP0l7O9VP3573tGYmNQhlhvZbFQWE3Yx7FkcNPQOZc+PopAjxW
9/s7iFeUy/yGeotYHKbMa5dch1XdbEwttskof6+BZNIS7nc5FSN70+Y2lLy36eCU
Om2xRLixTgWsZIbYXDSld2oeODcDUIgu/eS5klKXUme9yBoWULvYp0mYFQIDAQAB
o4ICHzCCAhswDgYDVR0PAQH/BAQDAgXgMAwGA1UdEwEB/wQCMAAwHwYDVR0jBBgw
FoAU6mujucETl34b+zqNaGAHOV+HSPowggG5BgNVHREEggGwMIIBrKBQBgorBgEE
AQkVAwQCoEITQDhCQzZEQTQzQkVFOUEzMUY2RkVCQUIxRDk3NERBMEZFNDFEMTMy
RjM2ODhFNTA4QkU2MzhGQTU3QjU1NjlFRjKgcAYKKwYBBAEJFQMEBKBiE2AwRkQw
QTAzNzA4MDUyODMxREM2RThGRkE1MzE1NjhCMEFGM0ZFQTRENEIxMTgwRUFFRUZF
MUQ4RjVBODlGNkVGOTQxMDE3MThGNjBBNkM1MDUwODgzNkY1MTlGOTFFMEOgUAYK
KwYBBAEJFQMEA6BCE0AwMjRCNENDNDgwRjE4Njg4MDAwMDAwMDAwMDAwMDAwMDAw
MDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwoFAGCisGAQQBCRUDBAOgQhNA
QzA0QjRDQzQ4MEYxODY4ODAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAw
MDAwMDAwMDAwMDAwMKBCBgkrBgEEAQkVAgOgNRMzQ2hpcElEPWhtVzVXR2hu
STRVZWpNSTRIeFRSaXdBQUFBQUFBQUFBQUFBQUFBQUE9MB0GA1UdDgQWBBQF
WtZgjSA/BQT55wuhOk6qq2vGyjANBgkqhkiG9w0BAQsFAAOCAQEAOtAl+Cxyn/lO
h2QDbW6Ors4xPrxADdWDgpdaH6q3qTmU/HRx6mv5T5/y0KRyrwCkFSQRTYBw1/De
7MPV9XXBhUWDAolWN4eqYTMZQF9KTdthmfmTZMQlRZpRTWWYWASfVyt0eObi/RNK
lGKlKyDA+uSfOyKUj6+Yays/bF4A5Bc0bORyCZ0asy4URQ9es6niS6/YuX/FAVrl
xl0BMuiSs83kXxk3lyIUbalN6xz4U17m83m58OWV1sTykRuS58NDyxhmtaWOFLYj
Zap4lJvzu7Dnk0pAgKvwxeUDT1wOJGij/NemwMgXzZEJ7tKgNCaMOpoYGWDPVIZz
pHF1L0A1eQ==
-----END CERTIFICATE-----
Signature version: 1
Signature:
[redacted]
```

オプションの RSA 2048 署名は、3 つの証明書、署名のバージョンおよびユーザーにより提供されるナンスに対するものです。

```
RSA PKCS#1v1.5 Sign {<Nonce (UINT64)> || <Signature Version (UINT32)> || <Cisco Root CA
2048 cert (DER)> ||
<Cisco subordinate CA (DER)> || <SUDI certificate (DER)> }
```

シスコの管理ソリューションには、上記の出力を解釈する機能が装備されています。しかし、OpenSSL コマンドを使用した簡単なスクリプトを使用してプラットフォームの ID を表示して署名を確認することもでき、それによってシスコの一意のデバイス ID を確保できます。

```
linux-host:~]openssl x509 -in sudicert.pem -subject -noout
subject= /serialNumber=PID:C9300-24P SN:FDO1946BG05/O=Cisco/OU=ACT-2 Lite SUDI/CN=C9300-24P
```

ソフトウェア整合性の確認

次に、ブート段階のチェックサム レコードを表示する例を示します。ハッシュ測定値は、連続してブートされたソフトウェアの 3 つの段階それぞれについて表示されます。これらのハッシュをシスコが提供する基準値と比較できます。出力に署名するオプションを使用すると、出力は正規であり改ざんされていないことを保証する機能が検証ツールに付与されます。リプレイ攻撃から保護するために、ナンスを提供できます。

（注） ブート整合性ハッシュはMD5ハッシュではありません。バンドルファイルに対して **verify /md5 cisco9k_iosxe.17.18.01.SPA.bin** コマンドを実行すると、ハッシュは一致しません。

インストールモードでの **show platform integrity sign nonce 123** コマンドの出力例を以下に示します。

この出力には、インストールされている各パッケージファイルの測定値が含まれます。

```
Device# show platform integrity sign nonce 123
Platform: C9350-48TX
Boot 0 Version: MA1007R07.1012142023
Boot 0 Hash: 6F213D15A4E5FAE794A18DF5984F2D1BC79FA6092B624974179C7FD106B97D8D
Boot Loader Version: System Bootstrap, Version 17.15.0.14r, DEVELOPMENT SOFTWARE
Boot Loader Hash: 7A237F1AD265BAE9BFE72741E7BA8231D67984F86D87382A29FC11BCC47CF30C
OS Version: 17.18.01
OS Hashes:
cisco9k_iosxe.17.18.01.SPA.bin:
63249C93EC1531C85D1DDB77886DB97336A35FFFC3B80C048C17442F0363AE8CCBB43F4178B88FE4611F1AC764BFD6CC9F4084CE8098663666B926AE6CB11709
cisco9k-wlc.17.18.01.SPA.pkg:
5FA17640BE34A0627588F21086FD0D0FC9FF4246266CF3A3E902EB3EF5D80BB566211771B72EB55D7C2BBFE2FEC8C2F9A9B9FAD28BB67E16B2095372D2A857AE
cisco9k-lni.17.18.01.SPA.pkg:
6A4A400A26BB3E5B6962CA4624EBBCF86871359077EBDFAF60D7331DF8E8AA7293F108322E6EDD8201CB816D074F1368BF6FD7779FC8EE0A68527D4E9B0D6AE6
cisco9k-guestshell.17.18.01.SPA.pkg:
5C0E5ED9876B8A1DBF9AD945C4D600308AF2312CB540173DDD0F262E363DC944B5A202C0D27731473311983748770DCCA2895B1C8CAEFB64F0BBF69720F6E084F
cisco9k-cc_srdriver.17.18.01.SPA.pkg:
263445E41F41A6867D1B7851E1F8CA0950C5F4347062A5850618C883A07C9CD81B8748BB4236C2EF15A71D4BD1F3E86F79EBACF5EFD18708BF0A1FB48547410D
cisco9k-webui.17.18.01.SPA.pkg:
AC669D965058EC7C708DA1FE396E292157A9151A08CBD42C4FC9C6FC1DE00FAFF1CE1735CCC87CAF4C19E615227FA0360B8FFC9B7C91A29C8F480752BD86A7CA
cisco9k-srdriver.17.18.01.SPA.pkg:
E371A47E8C315A2AEA55C086A2D0A13FB08C60F63853CFC192E71523C75C29BA9A3BF4478DA49648692DBF18144FAA8593EA07D39AAA387D45F8E59A5FBB60E4
cisco9k-rpbase.17.18.01.SPA.pkg:
2AB27567D8DE97623A2EE73E92519BD1DC2BFFAB9511B0CBC9A0FC60871E336E1E9FCB512005C7A5FD0468EEE8EB6C86219BC62B9939B96E50BC624498128093
PCR0: 72E291B753C405FAC5857969F1414DF0265F3BF6AA697E1A3EF67166DB5F8E6D
PCR8: 89AE6C797F6222869E389D2A4625EA854816FD432F501CB6091D4C467BEE8B49
Signature version: 1
Signature:
[redacted]
```

イメージ署名の確認

次の例は、SHA-512 ハッシュを使用した、ブートアップ時のイメージのセキュアコード署名チェックを表しています。

```
switch: boot
system has been configured to boot from device configuration.
boot: attempting to boot from [flash:cisco9k_iosxe.17.18.01.SPA.bin]
boot: reading file cisco9k_iosxe.17.18.01.SPA.bin
[redacted]


Loading image in Verbose mode: 1



Image Base is: 0x100000000
Image Size is: 0x45556D0D
Package header rev 3 structure detected
Package type:30000, flags:0x0
IsoSize = 1103171584
Parsing package TLV info:
000: 000000090000001D4B45595F544C565F -         KEY_TLV_
010: 5041434B4147455F434F4D5041544942 - PACKAGE_COMPATIB
020: 494C495459000000000000090000000B - ILITY
030: 4652555F52505F545950450000000009 - FRU_RP_TYPE
040: 000000184B45595F544C565F5041434B -     KEY_TLV_PACK
050: 4147455F424F4F544152434800000009 - AGE_BOOTARCH
060: 0000000E415243485F693638365F5459 -     ARCH_i686_TY
```

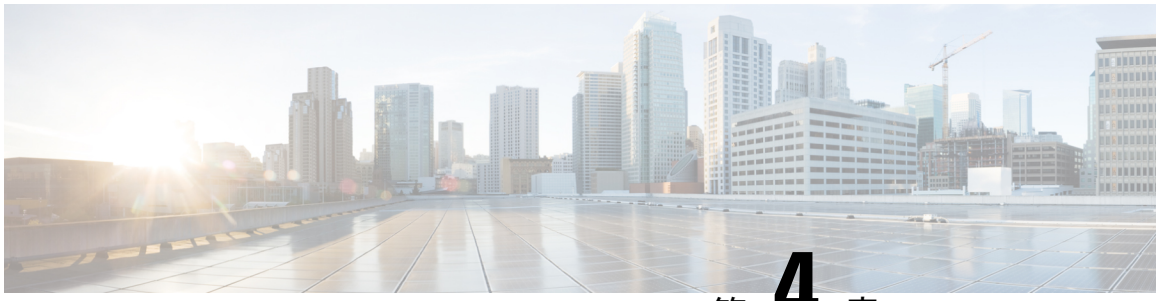
```
070: 5045000000000009000000144B45595F - PE          KEY_
080: 544C565F424F4152445F434F4D504154 - TLV_BOARD_COMPAT
090: 0000000900000012424F4152445F6369 -         BOARD_ci
0A0: 73636F396B5F5459504500000000009 - sco9k_TYPE
0B0: 000000184B45595F544C565F43525950 -     KEY_TLV_CRYP
0C0: 544F5F4B4559535452494E4700000009 - TO_KEYSTRING

TLV: T=9, L=29, V=KEY_TLV_PACKAGE_COMPATIBILITY
TLV: T=9, L=11, V=FRU_RP_TYPE
TLV: T=9, L=24, V=KEY_TLV_PACKAGE_BOOTARCH
TLV: T=9, L=14, V=ARCH_i686_TYPE
TLV: T=9, L=20, V=KEY_TLV_BOARD_COMPAT
TLV: T=9, L=18, V=BOARD_cisco9k_TYPE
TLV: T=9, L=24, V=KEY_TLV_CRYPTO_KEYSTRING
TLV: T=9, L=10, V=EnCrYpTiOn
TLV: T=9, L=11, V=CW_BEGIN=$$
TLV: T=9, L=19, V=CW_FAMILY=$cisco9k$
TLV: T=9, L=41, V=CW_IMAGE=$cisco9k_iosxe.17.18.01.SPA.bin$
TLV: T=9, L=21, V=CW_VERSION=$17.18.01$
TLV: T=9, L=50, V=CW_FULL_VERSION=$17.18.01.0.163.1754452537..IOSXE$
TLV: T=9, L=52, V=CW_DESCRIPTION=$Cisco IOS Software, IOS-XE Software$
TLV: T=9, L=9, V=CW_END=$$
Found DIGISIGN TLV type 12 length = 392
RSA Self Test Passed

Expected hash:
DDAF35A193617ABACC417349AE204131
12E6FA4E89A97EA20A9EEEE64B55D39A
2192992A274FC1A836BA3C23A3FEEBBD
454D4423643CE80E2A9AC94FA54CA49F

Obtained hash:
DDAF35A193617ABACC417349AE204131
12E6FA4E89A97EA20A9EEEE64B55D39A
2192992A274FC1A836BA3C23A3FEEBBD
454D4423643CE80E2A9AC94FA54CA49F
Sha512 Self Test Passed
CiscoRommonBootValidateTlv: Arch
Found package arch type ARCH_i686_TYPE
CiscoRommonBootValidateTlv: Fru
Found package FRU type FRU_RP_TYPE
CiscoRommonBootValidateTlv: Sha1
Performing Integrity Check ...
Returning default key storage: TAM

RSA Signed RELEASE Image Signature Verification Successful.
!
!
!
<<output truncated>>
```



第 4 章

同意トークン

同意トークン

同意トークンは、ネットワーク管理者と Cisco Technical Assistance Centre（Cisco TAC）の相互の同意により、システムシェルにアクセスする組織のネットワーク管理者を認証するために使用されるセキュリティ機能です。

一部のデバッグシナリオでは、Cisco TAC エンジニアが特定のデバッグ情報を収集したり、実稼働システムでライブデバッグを実行する必要がある場合があります。このような場合、Cisco TACエンジニアは、デバイスのシステムシェルにアクセスするようユーザー（ネットワーク管理者）に依頼します。同意トークンは、システムシェルへの特権による制限付きのセキュアなアクセスを提供する、ロック、ロック解除、および再ロックのメカニズムです。

同意トークン承認の仕組み

process_summary

システムシェルへのアクセスを要求する場合は、認証を受ける必要があります。最初にコマンドを実行し、デバイスの同意トークン機能を使用してチャレンジを生成する必要があります。デバイスは、一意のチャレンジを出力します。このチャレンジ文字列をコピーし、電子メールまたはインスタントメッセージでシスコ認定担当者に送信する必要があります。

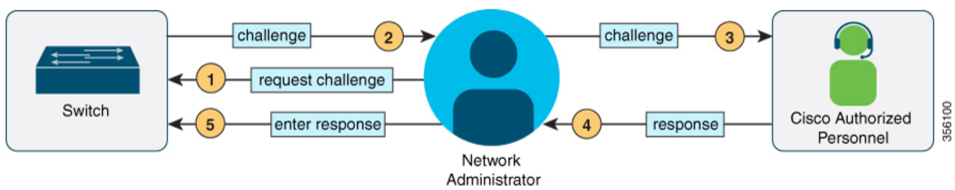
シスコ認定担当者は、一意のチャレンジ文字列を処理して一意のレスポンスを生成します。シスコ認定担当者はこのレスポンス文字列をコピーし、電子メールまたはインスタントメッセージで送信します。

次に、このレスポンス文字列をデバイスに入力する必要があります。チャレンジ/レスポンスペアが一致すると、システムシェルへのアクセスが許可されます。一致しない場合は、エラーが表示され、認証プロセスを繰り返す必要があります。

システムシェルにアクセスしたら、Cisco TAC エンジニアが必要とするデバッグ情報を収集します。システムシェルへのアクセスが完了したら、セッションを終了し、デバッグを続行します。

process_workflow

図 *1* : 同意トークン



1. 指定された期間だけシステムシェルへのアクセスを要求するチャレンジを生成します。
2. シスコ認定担当者にチャレンジ文字列を送信します。
3. デバイスにレスポンス文字列を入力します。
4. セッションを終了します。

システムシェルアクセスの同意トークン承認プロセス

システムシェルにアクセスするための同意トークン承認プロセスの詳細手順を以下に示します。

手順

ステップ 1 **request consent-token generate-challenge shell-access** *time-validity-slot*コマンドを使用して、チャレンジの要求を送信します。

例 :

```
Device# request consent-token generate-challenge shell-access auth-timeout 900
zSSdrAAAAQEBAQAAAABAgAEAAAAAAAMACH86csUhmD10BAAQ0FvdJ7CxqRYUeoD7B4AwW7QUABAAAAG8GAAhDVEFfREVNTwAGNUQ9ERU1EX0NUQ9TSUd0SU5HX0tFWQgAC0M5ODAwLUNMUs5CQALOvpQUEVSE5KrkI=
Device#
*Jan 18 02:47:06.733: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (challenge generation
 attempt: Shell access 0).
```

time-validity-slot : システムシェルへのアクセスを要求する期間（分単位）です。

この例では、セッションは 900 分後に期限切れになります。

デバイスが、base-64 形式の一意のチャレンジを生成します。

ステップ 2 デバイスによって生成されたチャレンジ文字列を、電子メールまたはインスタントメッセージでシスコ認定担当者に送信します。

シスコ認定担当者は、一意のチャレンジ文字列を処理してレスポンスを生成します。レスポンスもまた、固有のbase-64文字列です。シスコ認定担当者はこのレスポンス文字列をコピーし、電子メールまたはインスタントメッセージで送信します。

ステップ3 シスコ認定担当者から送信されたレスポンス文字列を、**request consent-token accept-response shell-access** *response-string* コマンドを使用して入力します。

例：

```
Device# request consent-token accept-response shell-access
[redacted]
% Consent token authorization success
*Jan 18 02:51:37.807: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authentication success:
 Shell access 0).

Device# request platform software system shell
Activity within this shell can jeopardize the functioning of the system.
Are you sure you want to continue? [y/n] y
Device#
*Jan 18 02:56:59.714: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (authorization for
Shell access 0 will expire in 10 min).
```

チャレンジ/レスポンスペアが一致すると、システムシェルへのアクセスが許可されます。チャレンジとレスポンスのペアが一致しない場合は、エラーが表示されます。手順1～3を繰り返して再試行します。

承認されると、要求されたタイムスロットのシステムシェルにアクセスできます。

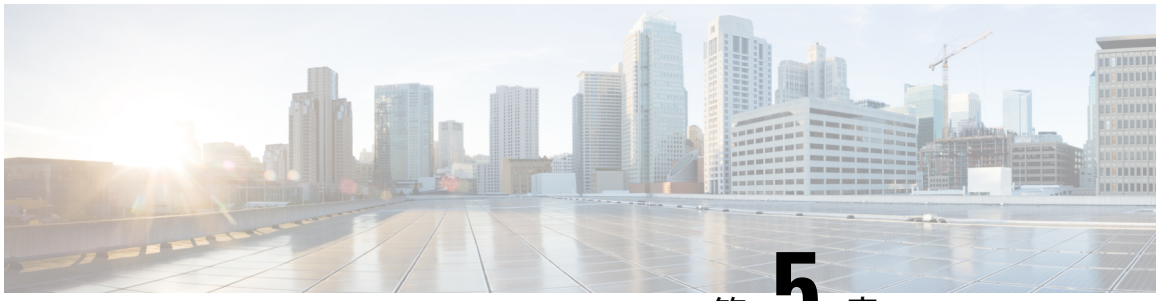
承認セッションの残り時間が10分になると、デバイスからメッセージが通知されます。

ステップ4 システムシェルへのアクセスが終了したら、**request consent-token terminate-auth** コマンドを使用してセッションを終了できます。承認セッションがタイムアウトする前に、このコマンドを使用してセッションを強制終了することもできます。要求したタイムスロットが期限切れになると、セッションも自動的に終了します。

例：

```
Device# request consent-token terminate-auth
% Consent token authorization termination success

Device#
*Jan 18 23:33:02.937: %CTOKEN-6-AUTH_UPDATE: Consent Token Update (terminate
authentication: Shell access 0).
Device#
```



第 5 章

工場出荷時の状態へのリセット

工場出荷時の状態へのリセット

工場出荷時設定へのリセットを行うと、デバイスに保存されているお客様特有のデータがすべて消去され、デバイスは出荷時の元の設定に復元されます。このプロセスでは、設定、ログファイル、ブート変数、コアファイル、およびクレデンシャル（連邦情報処理標準（FIPS）関連キーも含む）が消去されます。工場出荷時設定へのリセット時に行われるデータ消去は、NIST Special Publication 800-88 Revision 1で規定されている *Clear* 方式に適合しています。

工場出荷時の状態へのリセットをいつ行うか

工場出荷時の状態へのリセットは、次のシナリオで行う必要があります。

- デバイスの返品許可（RMA）

  RMA でデバイスをシスコに返す際は、RMA 証明書を取得する前に、お客様固有のすべてのデータを消去します。

- 侵害されたデバイスの回復

  デバイスに格納されているキーマテリアルまたはクレデンシャルが侵害された場合は、デバイスを工場出荷時の状態にリセットしてから、デバイスを再設定します。

工場出荷時の状態へのリセット時の動作

1. 初期設定へのリセット時、デバイスはリロードされ、ROMMON モードを開始します。

   初期設定へのリセット後、デバイスは、ソフトウェアの検索とロードに必要な **MAC_ADDRESS** 変数と **SERIAL_NUMBER** 変数を含むすべての環境変数を削除します。

2. ROMMON モードでリセットを実行すると、環境変数は自動的に設定されます。

   BAUD rate 環境変数は、初期設定へのリセット後にデフォルト値に戻ります。BAUD rate と console speed が常に同じであることを確認してください。同じでない場合、コンソールは応答しなくなります。
3. ROMMON モードでのシステムリセットが完了したら、USB または TFTP を使用して Cisco IOS XE ソフトウェアを追加します。

次の表に、初期設定へのリセットプロセス中に消去および保持されるデータの詳細を示します。

表 *3:* 初期設定へのリセット時に消去されるデータと保持されるデータ

消去されるデータ	保持されるデータ
すべての Cisco IOS XE ソフトウェアイメージ（現在のブートイメージも含む）	リモート Field-Replaceable Unit（FRU）からのデータ
クラッシュ情報とログ	コンフィギュレーション レジスタの値
ユーザーデータ、スタートアップおよび実行コンフィギュレーション、および Serial Advanced Technology Attachment（SATA）、SSD、USB などのリムーバブル ストレージ デバイスの内容	—
FIPS 関連キーなどのクレデンシャル	セキュアな固有デバイス識別子（SUDI）証明書、公開キーインフラストラクチャ（PKI）キーなどのクレデンシャル
オンボード障害ロギング（OBFL）ログ	—
ユーザーが追加した ROMMON 変数	—
ライセンス	—

セキュアデータワイプ

デバイスでは、ソフトウェアイメージ、デバイス設定、ソフトウェアログ、および操作履歴が保存されます。これらのエリアにはお客様特有のデータが存在する可能性があり、その情報には、お客様によって実装されたネットワークアーキテクチャおよび設計に関する詳細が含まれていることがあります。

セキュアデータワイプを行う方法

セキュアデータワイプを行うには、**factory-reset** コマンドで **all secure** オプションを使用します。これにより、データのサニタイズが行われ、デバイスが安全にリセットされます。デバイスのブートイメージが保持されます。

データのサニタイズが完了すると、デバイスがリロードされ、フラッシュからのイメージを使用してブートしていた場合は、デバイスイメージがフラッシュに保持されます。

セキュアデータワイプの標準

セキュアデータワイプ機能では、NIST SP 800-88 Rev. 1 に記述されているメディアサニタイズガイドラインが使用されています。NIST 800-88 は、米国立標準技術研究所（NIST）によって公開されている標準であり、メディアサニタイズに関するガイドラインを提供します。

NIST 800-88 内の PURGE 規格では、実験技術を使用してストレージメディア上のデータを回復不能にする方法が指定されています。NIST 800-88 PURGE 方式を使用してデバイスがサニタイズされていると、単純な非侵害データ復元技術や高度な実験技術ではデータを復元できません。

工場出荷時の状態へのリセットを行うためのガイドライン

- 工場出荷時の状態へのリセットプロセスを開始する前に、すべてのソフトウェアイメージ（現在のイメージも含む）、設定、および個人データをバックアップします。
- 工場出荷時設定へのリセットプロセスの進行中に電源が中断されないようにします。
- 工場出荷時設定へのリセットプロセスを開始する前に、すべての In-Service Software Upgrade（ISSU）の手順を完了しておきます。
- 工場出荷時設定へのリセットを行うと、インストール済みのソフトウェアパッチは復元されません。
- VTY セッションを介して **factory-reset** コマンドが発行された場合、初期設定へのリセットプロセスの完了後にセッションは復元されません。
- スイッチがスタック構成であるか、または Stackwise Virtual Link（SVL）モードの場合、**factory-reset** コマンドの **config** キーワードはサポートされません。
- 高可用性（HA）モードで構成されているモジュラ型シャーシデバイスの場合は、各スーパーバイザモジュールに工場出荷時設定へのリセットを適用する必要があります。

工場出荷時の状態へのリセット

工場出荷時の状態へのリセットを行うには、次の手順に従います。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

プロンプトが表示されたらパスワードを入力します。

ステップ 2 スイッチ設定に応じて、次のいずれかのコマンドを入力します。

- スイッチ

  factory-reset {**all** [**secure**] [**3-pass**] | **config** | **boot-vars**}

- スイッチ スタック

  factory-reset {**all** [**secure 3-pass**] | **config** | **boot-vars** | **switch** {*switch-number* | **all** {**all** [**secure 3-pass**] | **config** | **boot-vars**}}

例：

```
Device# factory-reset all
OR
Device# factory-reset all secure
```

デバイスを出荷時の設定にリセットします。

（注）
factory reset コマンドを使用するために必要なシステム設定はありません。

- **all**：NVRAMのすべての内容、現在のブートイメージ、ブート変数、起動設定データと実行設定データ、およびユーザーデータも含めて、すべての イメージを消去します。このオプションを使用することを推奨します。
- **all secure**：データのサニタイズを実行し、デバイスを安全にリセットします。

  （注）
  - **all secure** オプションを使用できるのはスタンドアロンデバイスに対してのみです。

    このオプションは、NIST SP 800-88 Rev. 1 で説明されているメディアサニタイズのガイドラインを実装します。

  - **factory-reset all secure** コマンドは、データのサニタイズを開始します。デバイスのブートイメージが保持されます。

    データのサニタイズが完了すると、デバイスがリロードされ、フラッシュからのイメージを使用してブートしていた場合は、デバイスイメージがフラッシュに保持されます。

- **secure 3-pass**：3-pass 上書きでデバイスからすべての内容を消去します。

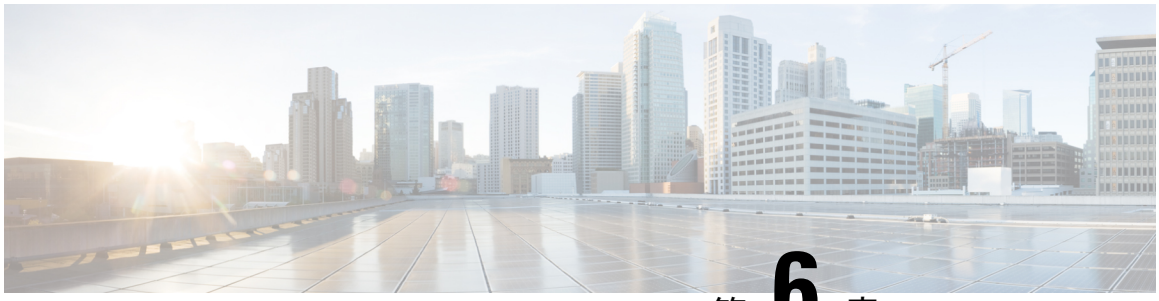
- Pass 1：すべてのアドレス可能な場所を 2 進数のゼロで上書きします。
- Pass 2：すべてのアドレス可能な場所を 2 進数の 1 で上書きします。
- Pass 3：すべてのアドレス可能な場所をランダムビットパターンで上書きします。

（注）
このオプションは、他のオプションの実行にかかる時間の約 3 倍の時間がかかります。

- **config**：スタートアップ コンフィギュレーションをリセットします。
- **boot-vars**：ユーザーによって追加されたブート変数を消去します。
- **switch** {*switch-number* | **all**}：
  - *switch-number*：スイッチ番号を指定します。
  - **all**：スタック内のすべてのスイッチを選択します。

工場出荷時の状態へのリセットプロセスが正常に完了すると、デバイスがリブートして ROMMON モードになります。

工場出荷時の状態へのリセット



第 6 章

セキュアなストレージ

セキュアなストレージ

セキュアストレージ機能では、重要な設定情報を暗号化することによってそれらが保護されます。非対称キーペア、事前共有秘密、タイプ 6 のパスワード暗号化キー、および特定のクレデンシャルなどが入っています。セキュリティを確保するため、インスタンスに特有の暗号化キーがハードウェア トラスト アンカー内に格納されているため、不正アクセスや侵害を防止できます。

セキュアストレージの有効化

セキュアストレージはデフォルトでは無効になっています。セキュアストレージを有効にするには、次の操作を行います。

手順

ステップ 1 **enable**

例：

```
Device> enable
```

特権 EXEC モードを有効にします。

プロンプトが表示されたらパスワードを入力します。

ステップ 2 **configure terminal**

例：

```
Device# configure terminal
```

グローバル コンフィギュレーション モードを開始します。

ステップ 3 **[no] service private-config-encryption**

例：

```
Device(config)# service private-config-encryption
```

デバイスでセキュアストレージ機能を有効にします。

デバイスでセキュアストレージを無効にするには、このコマンドの no 形式を使用します。

（注）
セキュアストレージを無効にすると、すべてのユーザーデータがプレーンテキストでNVRAMに保存されます。

ステップ 4 **end**

例：

```
Device(config)# end
```

特権 EXEC モードに戻ります。

ステップ 5 **write memory**

例：

```
Device# write memory
```

private-config ファイルを暗号化し、暗号化フォーマットで保存します。

暗号化ステータスを確認するための設定例

暗号化ステータスが有効になっているかどうかを確認するための出力例を以下に示します。

```
Device# show parser encrypt file status
Feature:            Enabled
File Format:        Cipher text
Encryption Version: ver1
```

この例では、ファイル形式は 暗号テキスト として表示されています。これは、ファイルが暗号化されていて、セキュアストレージが有効になっていることを表しています。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。