アクセス、認証およびアカウンティング

概要

ユーザ アクセス、認可およびアカウンティング

Application Policy Infrastructure Controller（APIC）ポリシーは、Cisco Application Centric Infrastructure（ACI）ファブリックの認証、認可、およびアカウンティング（AAA）機能を管理します。ユーザ権限、ロール、およびドメインとアクセス権限の継承を組み合わせることにより、管理者は細分化された方法で管理対象オブジェクト レベルで AAA 機能を設定することができます。これらの設定は、REST API、CLI、または GUI を使用して実行できます。

（注） ログイン ドメイン名に 32 文字を超えることはできないという既知の制限があります。また、ログイン ドメイン名とユーザ名を合わせた文字数は 64 文字を超えることはできません。

マルチテナントのサポート

コア Application Policy Infrastructure Controller（APIC）の内部データ アクセス コントロール システムにより、マルチテナント分離が提供され、テナント間での個人情報の漏洩が防止されます。読み取り/書き込みの制約により、テナントによる他のテナントの設定、統計情報、障害、またはイベント データの参照が防止されます。管理者によって読み取り権限が割り当てられない限り、テナントはファブリックの設定、ポリシー、統計情報、障害、またはイベントの読み取りが制限されます。

ユーザ アクセス：ロール、権限、セキュリティ ドメイン

APIC では、ロールベース アクセス コントロール (RBAC) を介してユーザのロールに従ってアクセスが提供されます。Cisco Application Centric Infrastructure (ACI) ファブリック ユーザは、次に関連付けられています。

- 事前定義またはカスタム ロール。ユーザに割り当てられた 1 つ以上の権限のセットです。
- 権限のセット。ユーザがアクセスできる管理対象オブジェクト（MO）を決定します。
- ロールごとの権限タイプ：アクセスなし、読み取り専用、または読み取り/書き込み
- ユーザがアクセスできる管理情報ツリー (MIT) の一部を識別する 1 つ以上のセキュリティ ドメイン タグ

ロールと権限

権限はシステム内の特定の機能に対するアクセス権を制御します。ACI ファブリックは、管理対象オブジェクト（MO）レベルでアクセス権限を管理します。すべてのオブジェクトは、読み取り可能な権限のリストと、書き込み可能な権限のリストを保持しています。特定の機能に対応するすべてのオブジェクトには、その機能の読み取りまたは書き込みリストの権限が付与されます。オブジェクトは追加の機能に対応する場合があるため、そのリストには複数の権限が含まれている場合があります。権限を含むロールがユーザに割り当てられると、そのユーザには、読み取りリストが読み取りアクセスを指定する関連オブジェクトへの読み取りアクセス権が付与され、書き込みリストが書き込みアクセスを指定するオブジェクトへの書き込みアクセス権が付与されます。

たとえば、「fabric-equipment」は、物理ファブリック内の機器に対応するすべてのオブジェクトへのアクセスを制御する権限です。物理ファブリック内の機器に対応するオブジェクト（「eqptBoard」など）には、特権リストに「fabric-equipment」が含まれます。「eqptBoard」オブジェクトは、「fabric-equipment」権限の読み取り専用アクセスを許可します。「fabric-admin」などの権限「fabric-equipment」が割り当てられているユーザには、「eqptBoard」オブジェクトへの読み取り専用アクセスなど、これらの機器オブジェクトへのアクセス権が付与されます。

（注） 一部のロールには他のロールが含まれています。たとえば、テナント管理者、ファブリック管理者、アクセス管理者などの「-admin」ロールは、同じベース名を持つロールのグループです。たとえば、「access-admin」は「access-connectivity」、「access-equipment」、「access-protocol」、および「access-qos」のグループです。同様に、tenant-adminは「テナント」ベースのロールのグループで、fabric-adminは「ファブリック」ベースのロールのグループです。

「admin」ロールにはすべての権限が含まれます。

ロールと権限の詳細については、『APIC ロールと権限マトリクス』を参照してください。

セキュリティ ドメイン

セキュリティ ドメインは、ACI MIT オブジェクト階層の特定のサブツリーに関連付けられたタグです。たとえば、デフォルトのテナント「common」にはドメイン タグ `common` が付いています。同様に、特殊なドメイン タグ `all` の場合、MIT オブジェクト ツリー全体が含まれます。管理者は、MIT オブジェクト階層にカスタム ドメイン タグを割り当てることができます。たとえば、管理者は「solar」という名前のテナントにドメイン タグ「solar」を割り当てることができます。MIT 内では、特定のオブジェクトだけがセキュリティ ドメインとしてタグ付けできます。たとえば、テナントはセキュリティ ドメインとしてタグ付けすることができますが、テナント内のオブジェクトはできません。

（注） セキュリティドメインのパスワード強度パラメータは、**[Custom Conditions]** を作成するか、または提供されている **[Any Three Conditions]** を選択して設定できます。

ユーザを作成してロールを割り当てても、アクセス権は有効になりません。1 つ以上のセキュリティ ドメインにそのユーザを割り当てることも必要です。デフォルトでは、ACI ファブリックには事前作成された次の 2 つの特殊なドメインが含まれています。

- `All`：MIT 全体へのアクセスを許可
- `Infra`：ファブリック アクセス ポリシーなどの、ファブリック インフラストラクチャのオブジェクトおよびサブツリーへのアクセスを許可

（注） ユーザのクレデンシャルが許可しない管理対象オブジェクトの読み取り操作の場合、「DN/Class Unauthorized to read」ではなく「DN/Class Not Found」というエラーが返されます。ユーザのクレデンシャルが許可しない管理対象オブジェクトへの書き込み操作の場合、「HTTP 401 Unauthorized」というエラーが返されます。GUI では、ユーザのクレデンシャルが許可しないアクションの場合、表示されないか、またはグレー表示されます。

事前に定義された一連の管理対象オブジェクト クラスをドメインに関連付けることができます。これらのクラスがオーバーラップすることはできません。ドメインの関連付けをサポートするクラスの例：

- レイヤ 2 およびレイヤ 3 のネットワークで管理されたオブジェクト
- ネットワーク プロファイル (物理、レイヤ 2、レイヤ 3、管理など)
- Quality of Service（QoS）ポリシー

ドメインに関連付けることができるオブジェクトが作成されると、ユーザは、ユーザのアクセス権の範囲内でオブジェクトにドメインを割り当てる必要があります。ドメインの割り当てはいつでも変更できます。

仮想マシン管理 (VMM) ドメインがセキュリティ ドメインとしてタグ付けされている場合、セキュリティ ドメイン内のユーザは、対応するタグ付き VMM ドメインにアクセスできます。たとえば、solar という名前のテナントに sun というセキュリティ ドメインのタグが付いており、

VMM ドメインにも sun というセキュリティ ドメインのタグが付いている場合、solar テナント内のユーザは各自のアクセス権限に従って VMM ドメインにアクセスできます。

連続してログインに失敗した後のユーザーのロックアウト

4.2(4) リリースおよび 5.0(1) リリース以降、ユーザーが設定された回数のログイン試行に失敗すると、ユーザーがログインできないようにすることができます。特定の期間内にユーザーが何回ログインに失敗可能かを指定できます。ユーザーが何度もログインに失敗すると、そのユーザーは指定された期間ログインできなくなります。

この機能は、Cisco Application Centric Infrastructure（ACI）データベースにあるローカル ユーザーと、RADIUS、LDAP、TACACS+、DUO プロキシ、SAML、RSA などの外部認証サーバーで認証されたリモート ユーザーの両方の失敗したログイン試行をカウントします。1 つの外部認証サーバー タイプを使用して連続して認証に失敗したためにロックアウトされたリモート ユーザーは、すべての外部認証サーバー タイプからロックアウトされます。たとえば、RADIUS サーバーを使用してログインに失敗した後にロックアウトされたユーザーは、LDAP サーバーを使用しているときにもロックアウトされます。AAA サーバーが到達不能またはダウンしたために失敗した認証、または不正な SSH キーが原因で失敗した認証は、ユーザーのロックアウトにはカウントされません。この機能は、間違ったパスワードの入力のみを考慮します。

クラスタ内の 1 つの Cisco Application Policy Infrastructure Controller（APIC）ノードからロックアウトされたユーザーは、リーフ スイッチとスパイン スイッチを含む、クラスタ内のすべてのノードからロックアウトされます。Cisco ACI データベースに存在しないローカル ユーザーは、この機能によりロックアウトできません。

（注） CLI を使用してこの機能を設定できません。

アクセス権のワークフローの依存関係

Cisco Application Centric Infrastructure (ACI) RBAC のルールによって、ファブリック全体へのアクセスを有効にするか、一部へのアクセスに制限します。たとえば、ベア メタル サーバ アクセス用のリーフ スイッチを設定するには、ログインしている管理者が `infra` ドメインに対する権限を持っている必要があります。デフォルトでは、テナント管理者は `infra` ドメインに対する権限を持っていません。この場合、リーフ スイッチに接続されているベア メタル サーバの使用を計画しているテナント管理者は、そのために必要なすべての手順を実行することはできません。テナント管理者は、`infra` ドメインに対する権限を持っているファブリック管理者と連携する必要があります。ファブリック管理者は、テナント管理者が ACI リーフ スイッチに接続されたベア メタル サーバを使用するアプリケーション ポリシーを導入するために使用するスイッチ設定ポリシーをセットアップします。

AAA RBAC の役割および権限

Application Policy Infrastructure Controller（APIC）は、次の AAA ロールと権限を提供します。

（注） Cisco APIC リリース 5.0(1) では、関連する多くのレガシー権限が統合されているため、権限の数は以前のリリースから削減されています。以前の権限から現在の権限へのマッピングを レガシー権限の再マッピング に示します。

（注） Cisco APIC で定義された各ロールについて、APIC のロールと権限のマトリックスには、書き込み可能な管理オブジェクトクラスと読み取り可能な管理オブジェクトクラスが表示されます。

表 *1 :* ロールの権限：管理

ロール：管理	
特権	**説明**
admin	すべてのファブリックの機能へのフルアクセスを提供します。管理者権限は、その他のすべての権限を組み合わせたものとみなされます。

表 *2:* ロールの権限：*aaa*

ロール：**aaa**	
特権	**説明**
aaa	ポリシーの認証、許可、アカウンティング、インポート/エクスポートの設定に使用されます。

表 *3:* ロールの権限：*access-admin*

Role: access-admin	
特権	**説明**
access-connectivity	インフラでのレイヤ 1 ～ 3 の構成、テナントの L3Out でのスタティックルート構成、管理インフラポリシー、テナント ERSPAN ポリシーに使用されます。
access-equipment	アクセス ポート設定に使用されます。
access-protocol	インフラストラクチャ、NTP、SNMP、DNS、およびイメージ管理用のファブリック全体のポリシー、およびクラスタポリシーやファームウェアポリシーなどの操作関連のアクセスポリシーでレイヤ 1 ～ 3 のプロトコル構成に使用されます。
access-qos	CoPP および QoS に関連するポリシーの変更に使用されます。

表 *4:* ロールの権限：*fabric-admin*

ロール：**fabric-admin**	
特権	**説明**
fabric-connectivity	ファブリック、ファームウェア、および導入ポリシーのレイヤ 1 ～ 3 の構成に使用します。ポリシー導入の影響を推定するための警告、およびリーフスイッチとスパインスイッチのアトミックカウンタ、診断、およびイメージ管理ポリシーを生成します。
fabric-equipment	リーフ スイッチおよびスパイン スイッチのアトミック カウンタ、診断、およびイメージ管理ポリシーに使用されます。

ロール：**fabric-admin**	
特権	**説明**
fabric-protocol	ファブリックでのレイヤ 1 ～ 3 のプロトコル構成、NTP、SNMP、DNS、およびイメージ管理のファブリック全体のポリシー、ERSPAN およびヘルススコアポリシー、およびファームウェア管理のトレースルートおよびエンドポイント トラッキングポリシーに使用されます。

表 *5 :* ロールの権限：*nw-svc-admin*

ロール：**nw-svc-admin**	
特権	**説明**
nw-svc-policy	レイヤ 4 ～ レイヤ 7 ネットワークサービス オーケストレーションの管理に使用されます。

表 *6 :* ロールの権限：*nw-svc-params*

ロール： **nw-svc-params**	
特権	**説明**
nw-svc-params	レイヤ4～レイヤ7のサービス ポリシーの管理に使用されます。

表 *7 :* ロールの権限：*ops*

Role: ops	
特権	**説明**
ops	設定されているポリシーの表示に使用されます（ポリシーのトラブルシューティングなど）。 （注） **Ops** ロールは、新しいモニタリング ポリシーおよびトラブルシューティング ポリシーの作成には使用できません。これらのポリシーは、Cisco APIC の他のすべての構成と同様に、**admin** 権限を使用して作成する必要があります。

表 *8 :* ロールの権限：*port-mgmt*

ロール：**port-mgmt**	
特権	**説明**
port-mgmt	ノードをセキュリティドメインに割り当てるために使用されます。また、ノードルールを持つセキュリティドメインのユーザーは、`port-mgmt` のロールを持つドメイン `all` に割り当てる必要があります。

表 *9 :* ロールの権限：*tenant-admin*

Role: tenant-admin	
特権	**説明**
aaa	ポリシーの認証、許可、アカウンティング、インポート/エクスポートの設定に使用されます。
access-connectivity	インフラでのレイヤ 1 ～ 3 の構成、テナントの L3Out でのスタティックルート構成、管理インフラポリシー、テナント ERSPAN ポリシーに使用されます。
access-equipment	アクセス ポート設定に使用されます。
access-protocol	インフラストラクチャ、NTP、SNMP、DNS、およびイメージ管理用のファブリック全体のポリシー、およびクラスタポリシーやファームウェアポリシーなどの操作関連のアクセスポリシーでレイヤ 1 ～ 3 のプロトコル構成に使用されます。
access-qos	CoPP および QoS に関連するポリシーの変更に使用されます。
fabric-connectivity	ファブリック、ファームウェア、および導入ポリシーのレイヤ 1 ～ 3 の構成に使用します。ポリシー導入の影響を推定するための警告、およびリーフスイッチとスパインスイッチのアトミックカウンタ、診断、およびイメージ管理ポリシーを生成します。
fabric-equipment	リーフ スイッチおよびスパイン スイッチのアトミック カウンタ、診断、およびイメージ管理ポリシーに使用されます。
fabric-protocol	ファブリックでのレイヤ 1 ～ 3 のプロトコル構成、NTP、SNMP、DNS、およびイメージ管理のファブリック全体のポリシー、ERSPAN およびヘルススコアポリシー、およびファームウェア管理のトレースルートおよびエンドポイントトラッキングポリシーに使用されます。

Role: tenant-admin	
特権	説明
nw-svc-policy	レイヤ 4 ～ レイヤ 7 ネットワークサービス オーケストレーションの管理に使用されます。
ops	設定されているポリシーの表示に使用されます（ポリシーのトラブルシューティングなど）。 （注） **Ops** ロールは、新しいモニタリング ポリシーおよびトラブルシューティング ポリシーの作成には使用できません。これらのポリシーは、Cisco APIC の他のすべての構成と同様に、**admin** 権限を使用して作成する必要があります。
tenant-connectivity	ブリッジドメイン、サブネット、および VRF などの レイヤ 1 ～ 3 の接続変更に使用されます。これには、リーフスイッチおよびスパインスイッチのアトミックカウンタ、診断、およびイメージ管理ポリシー、テナントのインバンドおよびアウトオブバンド管理接続構成。アトミックカウンタやヘルススコアなどのデバッグ/モニタリングポリシーなどがあります。
tenant-epg	エンドポイント グループの削除/作成など、テナント構成の管理に使用されます。
tenant-ext-connectivity	ファームウェアポリシーの書き込みアクセスに使用されます。これには、テナント L2Out および L3Out 構成の管理、traceroute、ping、oam、eptrk などのデバッグ/モニタリング/オブザーバポリシーがあります。
tenant-ext-protocol	BGP、OSPF、PIM、IGMP などのテナント外部レイヤ 1 ～ 3 プロトコルの管理、およびトレースルート、ping、oam、eptrk などのデバッグ/モニタリング/オブザーバポリシーに使用されます。通常、ファームウェア ポリシーの書き込みアクセスにのみ使用します。
tenant-network-profile	ネットワーク プロファイルの削除および作成、エンドポイント グループの削除および作成など、テナント設定の管理に使用されます。
tenant-protocol	テナント下のレイヤ 1 ～ 3 プロトコルの構成、テナント トレースルートポリシー、およびファームウェアポリシーの書き込みアクセスに使用されます。
tenant-qos	テナントの QoS に関連する設定に使用されます。
tenant-security	テナントのコントラクトに関連する設定に使用されます。

Role: tenant-admin	
特権	説明
vmm-policy	認証や接続など、仮想マシンネットワークのポリシーを管理するために使用されます。

表 *10 :* ロールの権限：*tenant-ext-admin*

Role: tenant-ext-admin	
特権	説明
tenant-connectivity	ブリッジドメイン、サブネット、および VRF などの レイヤ 1 ～ 3 の接続変更に使用されます。これには、リーフスイッチおよびスパインスイッチのアトミックカウンタ、診断、およびイメージ管理ポリシー、テナントのインバンドおよびアウトオブバンド管理接続構成。アトミックカウンタやヘルススコアなどのデバッグ/モニタリングポリシーなどがあります。
tenant-epg	エンドポイント グループ、VRF、ブリッジ ドメインの削除/作成など、テナント設定の管理に使用されます。
tenant-ext-connectivity	ファームウェアポリシーの書き込みアクセスに使用されます。これには、テナント L2Out および L3Out 構成の管理、traceroute、ping、oam、eptrk などのデバッグ/モニタリング/オブザーバポリシーがあります。
tenant-ext-protocol	BGP、OSPF、PIM、IGMP などのテナント外部レイヤ 1 ～ 3 プロトコルの管理、およびトレースルート、ping、oam、eptrk などのデバッグ/モニタリング/オブザーバポリシーに使用されます。通常、ファームウェア ポリシーの書き込みアクセスにのみ使用します。
tenant-network-profile	ネットワーク プロファイルの削除および作成、エンドポイント グループの削除および作成など、テナント設定の管理に使用されます。
tenant-protocol	テナント下のレイヤ 1 ～ 3 プロトコルの構成、テナント トレースルートポリシー、およびファームウェアポリシーの書き込みアクセスに使用されます。
tenant-qos	テナントの QoS に関連する設定に使用されます。
tenant-security	テナントのコントラクトに関連する設定に使用されます。
vmm-policy	認証や接続など、仮想マシンネットワークのポリシーを管理するために使用されます。

表 *11 :* ロールの権限： *vmm-admin*

Role: vmm-admin	
特権	説明
vmm-policy	認証や接続など、仮想マシンネットワークのポリシーを管理するために使用されます。

カスタム ロール

カスタム ロールを作成し、ロールに権限を割り当てることができます。インターフェイスは、すべての管理対象オブジェクト クラスに 1 つ以上の権限を内部的に割り当てます。XML モデルで、権限はアクセス属性に割り当てられています。権限のビット数は、コンパイル時に割り当てられ、クラスのインスタンスまたはオブジェクトごとではなく、クラスごとに適用されます。

45 権限ビットだけでなく、「aaa」権限ビットはすべての AAA サブシステムの設定と読み取り操作に適用されます。次の表は、サポートされている権限の組み合わせの一覧を提供します。表の行は Cisco Application Centric Infrastructure（ACI）モジュールを表し、列は特定のモジュールの機能を表します。セルの「○」の値は、モジュールがアクセス可能な機能と、機能にアクセスするための権限ビットが存在することを示します。空のセルは、権限ビットでアクセスできないモジュールの特定の機能を示します。権限ビットについての詳細は、各ビットの機能について参照してください。

	Connectivity	**QoS**	セキュリティ	アプリケーション	**Fault**	**Stats**	**Provider**	サービス プロファイル	サービス チェーン
VMM	はい		はい		はい	はい	はい		
ファブリック	はい	はい	はい	はい	はい	はい	はい		
External	はい	はい	はい		はい	はい			はい
テナント	はい	はい	はい	EPG、NP	はい	はい			はい
Infra	はい	はい	はい	はい	はい	はい			はい
操作					はい	はい			

	Connectivity	QoS	セキュリティ	アプリケーション	Fault	Stats	Provider	サービス プロファイル	サービス チェーン
ストレージ	はい	はい	はい	はい	はい	はい			
ネットワーク サービス	はい	はい	はい	はい	はい	はい		はい	

複数のセキュリティ ドメイン間で物理リソースを選択的に公開する

ファブリック全体の管理者は、RBAC規則を使用して、異なるセキュリティドメインにあるため他の方法ではアクセス不可能なユーザに対し、物理リソースを選択的に公開します。

たとえば、ソーラーというテナントのユーザが仮想マシン管理（VMM）ドメインへのアクセスを必要とする場合、ファブリック全体の管理者によって、これを許可するRBAC規則を作成することができます。RBAC 規則は、次の 2 つの部分から構成されます。アクセス対象オブジェクトを検索する識別名（DN）と、オブジェクトにアクセスするユーザを含むセキュリティドメインの名前です。したがって、この例では、ソーラーというセキュリティドメイン内の指定ユーザがログインすると、このルールにより、VMMドメインおよびツリーの内の子オブジェクトすべてへのアクセスが許可されます。VMM ドメインへのアクセスを複数のセキュリティドメイン内のユーザに許可するには、ファブリック全体の管理者は、セキュリティドメインそれぞれについて、VMM ドメインの DN とセキュリティ ドメインを含む RBAC 規則を作成します。

（注） 管理情報ツリー内の異なる部分に存在するユーザに対し、RBAC規則によりオブジェクトを公開することは可能ですが、CLIの使用によってツリーの構造を横断することでそのようなオブジェクトに移動することはできません。ただし、RBAC規則に含まれるオブジェクトのDNをユーザが把握していれば、ユーザはMO検索コマンドにより、CLIを使用してそれを見つけることができます。

複数のセキュリティ ドメイン間でのサービス共有を有効にする

ファブリック全体の管理者は、RBAC規則を使用して、テナント間の共有サービスを可能にするトランステナント EPG 通信をプロビジョニングします。

APIC ローカル ユーザ

管理者は、外部AAAサーバを使用しないことを選択し、APIC自体でユーザを設定することができます。これらのユーザは、APIC ローカル ユーザと呼ばれます。

ユーザがパスワードを設定する時点で、APIC によって以下の基準が検証されます。

- パスワードの最小長は 8 文字です。

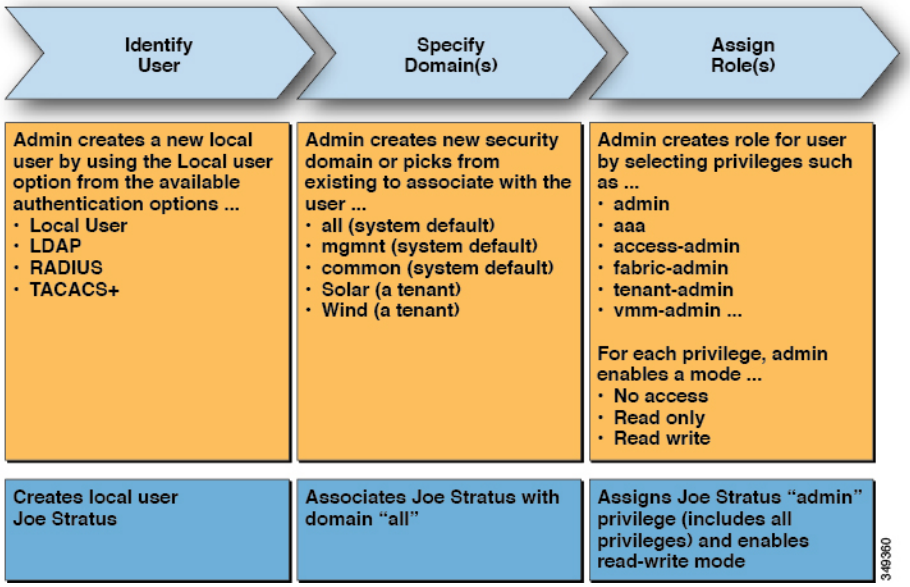
- パスワードの最大長は 64 文字です。
- 連続して繰り返される文字は 3 文字未満です。
- 小文字、大文字、数字、記号の文字種のうち少なくとも 3 種類の文字が含まれている必要があります。
- 簡単に推測できるパスワードは使用しません。
- ユーザ名やユーザ名を逆にしたものは使用できません。
- cisco、isco、またはこれらの文字列の並べ替えを変化させたものや、それらの文字の大文字化の変更により取得される変形語であってはなりません。

Cisco ACI では、パスワードの保存に SHA256 一方向ハッシュを使用した暗号化ライブラリが使用されます。保管中のハッシュされたパスワードは、暗号化されたファイルシステムに保存されます。暗号化されたファイルシステムのキーは、Trusted Platform Module（TPM）を使用して保護されます。

また APIC により、管理者は、外部で管理されている認証 Lightweight Directory Access Protocol（LDAP）、RADIUS、TACACS+、または SAML サーバで設定されたユーザにアクセス権を付与できます。ユーザは、異なる認証システムに属し、APIC に同時にログインできます。

次の図は、ACI ファブリック全体へのフル アクセス権があるローカル APIC 認証データベース内の管理ユーザを設定するプロセスがどのように動作するかを示します。

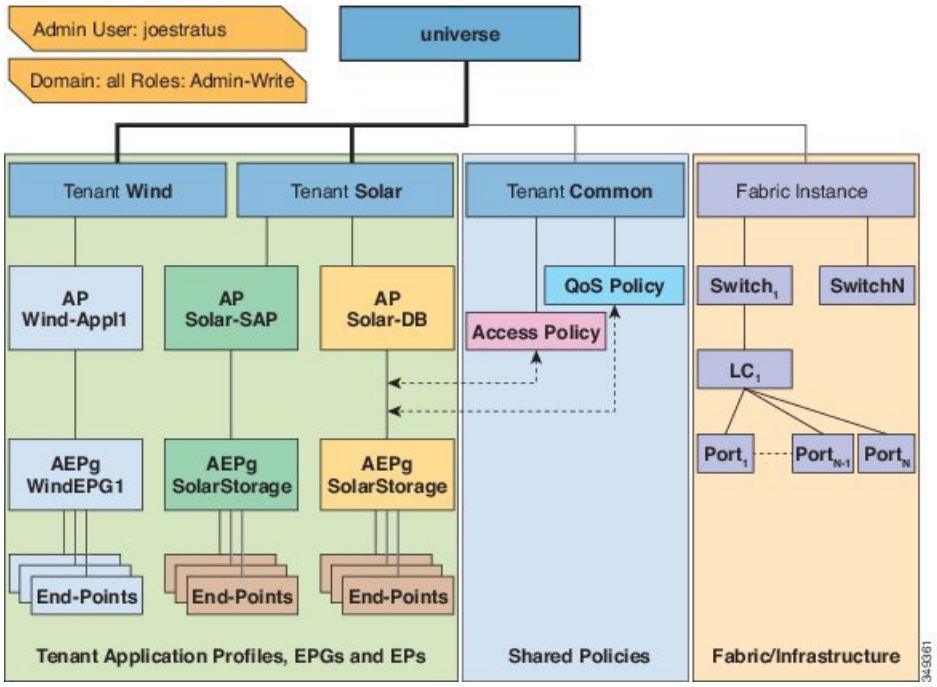
図 *1 : APIC* ローカル ユーザの設定プロセス



（注） セキュリティ ドメイン「all」は、管理情報ツリー（MIT）全体を表します。このドメインには、システム内のすべてのポリシーと APIC よって管理されるすべてのノードが含まれます。テナント ドメインには、テナントのすべてのユーザおよび管理対象オブジェクトが含まれます。テナント管理者には、「all」ドメインへのアクセス権を付与しないでください。

次の図は、管理ユーザ Joe Stratus が持つシステムへのアクセス権を示します。

図 *2*：「*all*」ドメインへ管理ユーザを設定した結果



読み取り/書き込み「管理者」権限を持つユーザ Joe Stratus は、ドメイン「all」に割り当てられ、システム全体へのフル アクセス権が与えられます。

ローカル ユーザー向け OTP ベース 2 要素認証

ファブリック管理者ユーザーは、ローカルユーザーのワンタイムパスワード（OTP）機能を有効にできます。ワンタイムパスワードは30秒ごとに変更され、セキュリティが強化されます。OTP を有効にすると、Cisco Application Policy Infrastructure Controller（APIC）は、base32 OTP キーである、ランダムな人間が判読できる 16 バイナリオクテットを生成します。この OTP キーは、二要素認証に使用されるユーザーの OTP を生成するために使用されます。

Cisco APIC は、二要素認証で使用する次のセキュリティ プラットフォームをサポートしています。

- Duo Mobile App を使用した Duo Security
- Google、Google Authenticator アプリ（Android および Apple iOS スマートフォンのみ）

（注） 対応しているアプリストアから、表示されたアプリをダウンロードする必要があります。

これらのセキュリティ プラットフォームは、ユーザー ID のリポジトリとして機能しません。これらのプラットフォームは、組織の既存の認証（オンプレミスまたはクラウドベース）に加えて、二要素認証を提供します。二要素認証は、ユーザーが組織のプライマリ認証ソースで認証を完了すると発生します。

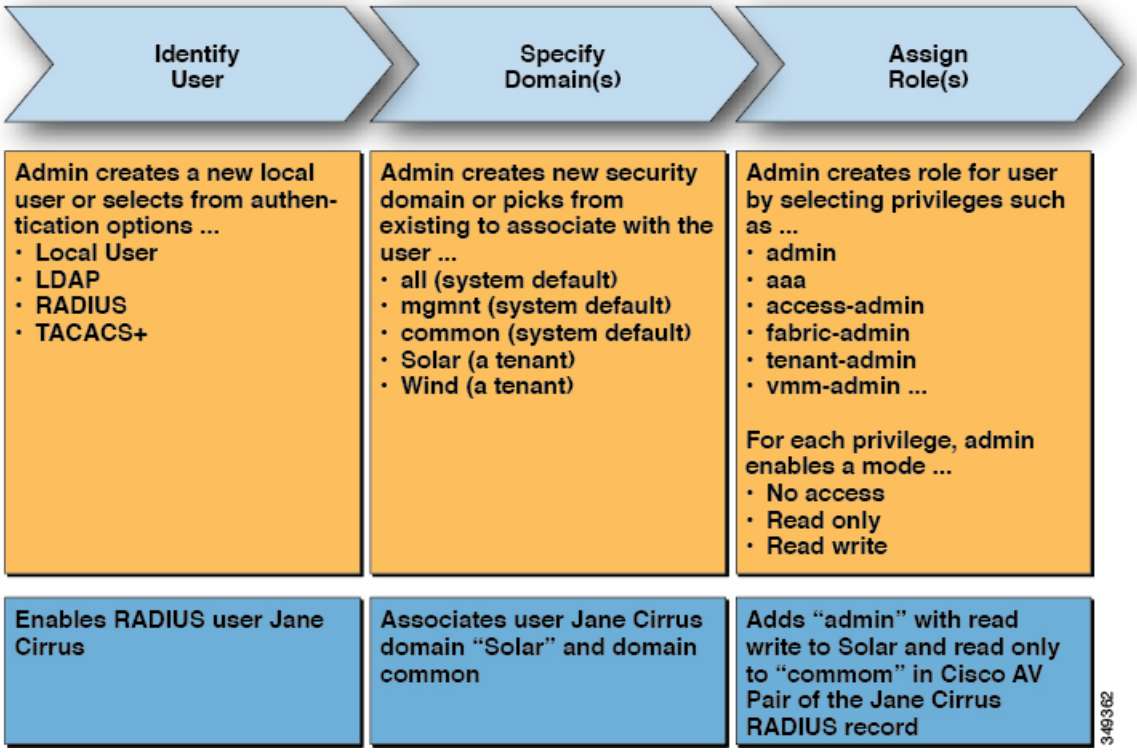
プラットフォームは、プライマリ認証ソースで認証を完了した後、3 種類の二要素認証方法をサポートします。

- スマートフォンで適切なモバイルアプリを使用したモバイルでのプッシュ通知。
- 登録済みの電話または携帯電話での通話。
- 適切なモバイルアプリで生成されるパスコード。

外部管理されている認証サーバのユーザ

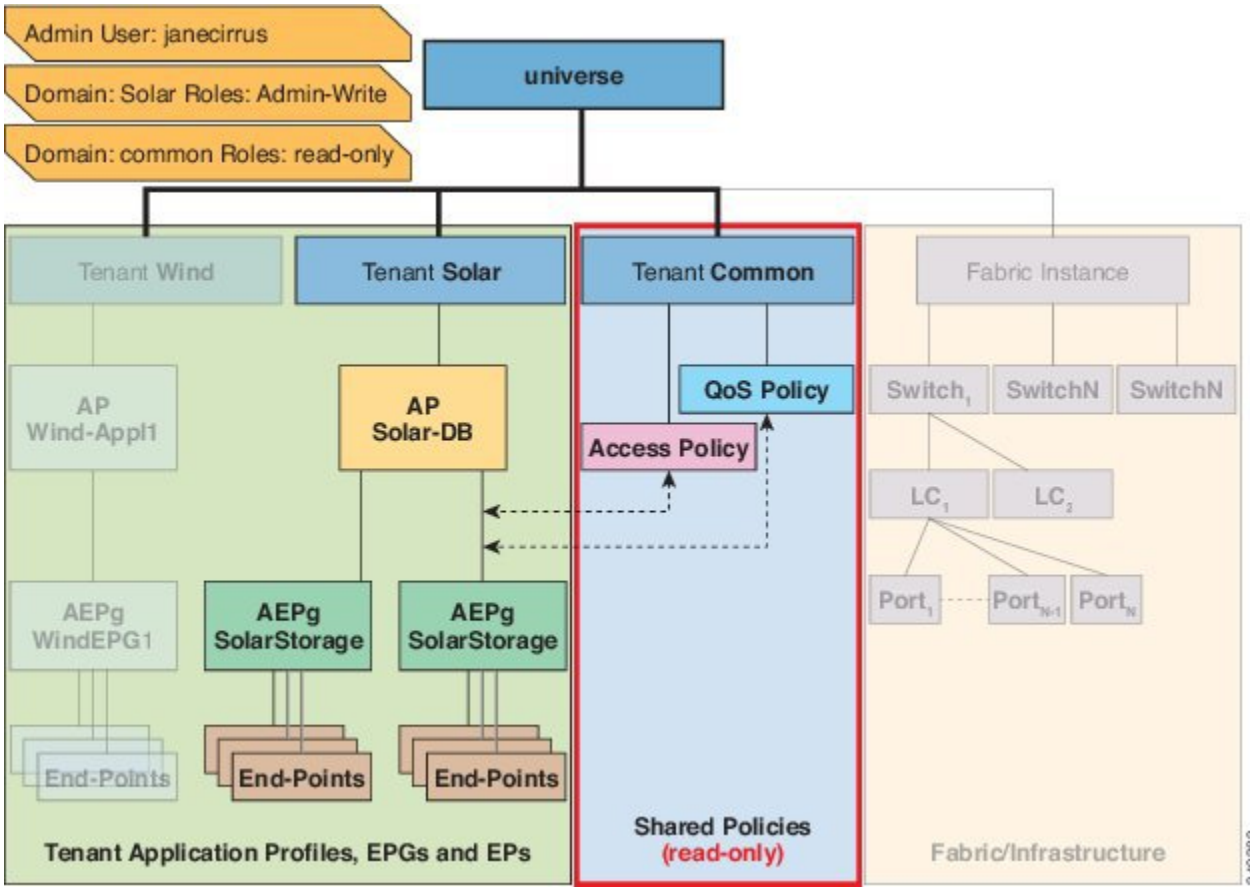
次の図は、テナント Solar へのフル アクセス権がある外部 RADIUS サーバ内の管理ユーザを設定するプロセスがどのように動作するかを示します。

図 *3*: 外部認証サーバでのユーザ設定のプロセス



次の図は、管理ユーザ Jane Cirrus が持つシステムへのアクセス権を示します。

図 *4:* テナント ***Solar*** へ管理ユーザを設定した結果



この例では、Solar テナントの管理者には、Solar テナントに含まれるすべてのオブジェクトへのフル アクセス権と、テナント Common への読み取り専用アクセス権があります。テナント管理者 Jane Cirrus には、テナント Solar へのフル アクセス権があり、テナント Solar で新しいユーザを作成する機能などがあります。テナント ユーザは、自身が所有し制御する ACI ファブリックの設定パラメータを変更できます。また、エンドポイント、エンドポイント グループ（EPG）およびアプリケーション プロファイルなどの適用されるエンティティ（管理対象オブジェクト）の統計情報の読み取り、障害およびイベントのモニタもできます。

上記の例では、ユーザ Jane Cirrus は外部 RADIUS 認証サーバで設定されました。外部認証サーバで AV ペアを設定するには、既存のユーザ レコードに Cisco AV ペアを追加します。Cisco AV ペアは、APIC 上のユーザに対するロールベース アクセス コントロール（RBAC）のロールと権限を指定します。次に RADIUS サーバは、ユーザ権限を APIC コントローラに伝播します。

上記の例のオープン RADIUS サーバ（/etc/raddb/users）の設定は次のとおりです。

```
janecirrus Cleartext-Password := "<password>"
Cisco-avpair = "shell:domains = solar/admin/,common//read-all(16001)"
```

この例には、次の要素が含まれています。

- `janecirrus` はテナント管理者です。

- `solar` はテナントです。
- `admin` は書き込み権限があるロールです。
- `common` は、テナント共通サブツリーで、すべてのユーザがそのサブツリーへの読み取り専用アクセス権を持っています。
- `read-all` は、読み取り権限があるロールです。

Cisco AV ペアの形式

Cisco APIC は、管理者が外部認証サーバで Cisco AV ペアを設定し、1 個の AV ペアの文字列のみを検索することを要求しています。これを行うには、管理者は既存のユーザ レコードに Cisco AV ペアを追加します。Cisco AV ペアは、ユーザの RBAC ロールおよび権限に必要な APIC を指定します。

AV ペア文字列を機能させるため、次の形式にする必要があります。

```
shell:domains =
ACI_Security_Domain_1/ACI_Write_Role_1|ACI_Write_Role_2|ACI_Write_Role_3/ACI_Read_Role_1|ACI_Read_Role_2,
ACI_Security_Domain_2/ACI_Write_Role_1|ACI_Write_Role_2|ACI_Write_Role_3/ACI_Read_Role_1|ACI_Read_Role_2,
ACI_Security_Domain_3/ACI_Write_Role_1|ACI_Write_Role_2|ACI_Write_Role_3/ACI_Read_Role_1|ACI_Read_Role_2
```

- **shell:domains=**：ACI が正常に文字列を読み取るために必要です。シェル文字列を常にプリペンドする必要があります。
- **ACI_Security_Domain_1//admin**：管理者にこのセキュリティ ドメインのテナントへの読み取り専用アクセス権を付与します。
- **ACI_Security_Domain_2/admin**：管理者にこのセキュリティ ドメインのテナントへの書き込みアクセス権を付与します。
- **ACI_Security_Domain_3/read-all**：このセキュリティ ドメインのテナントへの読み取り/書き込みすべてのアクセス権を付与します。

（注） **/** により区別される文字列のセキュリティ ドメイン、書き込み、読み取りセクション同じセキュリティ ドメイン内の **|** により区別される複数の書き込みまたは読み取り権限

（注） Cisco APIC リリース 2.1 より、AV ペアに UNIX ID が指定されていない場合、APIC は UNIX の固有ユーザー ID を内部的に割り当てます。

APIC は、次の正規表現をサポートしています。

```
shell:domains\\s*[=:]\\s*((\\S+?/\\S*?/\\S*?),\\S+?/\\S*?/\\S*?){0,31})(\\(\\d+\\))$
shell:domains\\s*[=:]\\s*((\\S+?/\\S*?/\\S*?),\\S+?/\\S*?/\\S*?){0,31})$
```

例：

- 例 1：writeRole のみを持つ単一のログイン ドメインを含む Cisco AV ペア

```
shell:domains=ACI_Security_Domain_1/Write_Role_1|Write_Role_2/
```

- 例 2：readRole のみを持つ単一のログイン ドメインを含む Cisco AV ペア

```
shell:domains=Security_Domain_1//Read_Role_1|Read_Role_2
```

（注） 文字「/」はログイン ドメインごとに writeRole と readRole の間を区切る記号で、使用するロールの種類が 1 つのみである場合も必要です。

Cisco AV ペアの文字列は、大文字と小文字が区別されます。エラーが表示されなくても、使用する大文字と小文字がドメイン名またはロールに一致していない場合は、予期しない権限が付与されることがあります。

AV ペア GUI の設定

セキュリティ ドメインは、**[Admin]** > **[AAA]** > **[Security Management]** > **[Security Domains]** の ACI GUI で定義されており、**[テナント]** > **[Tenant_Name]** > **[ポリシー]** のテナントに割り当てられています。

セキュリティ ドメインには読み取りまたは書き込み権限のいずれかが必須です。これらの権限は、**[APIC]** > **[Admin]** > **[Security Management]** > **[Roles]** で定義されています。権限が書き込みセクションに入力される場合、ACI_Security_Domain_1/admin/admin/admin を使用する必要がないため、自動的に同じレベルの読み取り権限を付与します。

リモート ユーザー ロールの変更

ユーザー権限を「動的」に変更可能で、ユーザーがロール変更の要求を行うことが可能になり、ローカルまたはリモートで保存されている情報に基づいて、要求ロールが許可または拒否されます。

ロール変更は Cisco ACS サーバー経由でのみサポートされており、明示的な「要求」に基づくロールの割り当てによって実行できます。

ACI ファブリックは、Radius、TACACS +、LDAP プロトコルを使用して外部認証をサポートします。上記の両方の方法で、リモート認証サーバにロール変更機能をサポートするコンポーネントが含まれていると仮定します。

Cisco Secure ACS サーバーは、TACACS+ プロトコルのリモート認証、認証、およびアカウンティングの機能を提供します。

デフォルト デバイス管理または**デフォルト ネットワーク アクセス サービス**のどちらかにルールが一致する必要があります。

認証で、別のルール設定が設定されています。

- **AVPairOps**：tacacs + ユーザー名および AVPair 値と一致します（cisco-av-pair*newrole）。ルールに一致すると、ACI_OPS シェル プロファイルが返されます

- **NoAVPair**：tacacs + ユーザー名のみ一致し、一致で ACI_ADMIN シェル プロファイルを返します
- **opsuser**：プロトコルのみ一致し、ACI_OPS シェル プロファイルを返します

GUI を使用したリモート ユーザー ロールの変更

始める前に

ロールは、最初に AVPair と一致するように Cisco ASC サーバーで設定し、その一致に基づいてシェル認証プロファイルとして選択する必要があります。

ステップ 1 ASC 認証ポリシーを作成します。**[Access Policies]** > **[Access Services]** > **[Default Device Admin Identity]** に移動し、次の手順を実行します。

（注） シェル プロファイルが CiscoAVPair を使用して設定され、ユーザの認証に使用されます。

a) [TACACS+:AVPair equals cisco-av-pair*] に条件を追加し、[OK] をクリックします。

（注） デフォルトでは、ユーザは **cisco-av-pair** ロールを使用して認証されます。

b) [TACACS+:AVPair equals cisco-av-pair*readall] に条件を追加し、[OK] をクリックします。

（注） APIC でキーワード **readall** を使用して、ロールを **default** ロールから **readall** ロールに変更します（シェル プロファイルで read-all が設定されます）。

ステップ 2 APIC GUI にログインし、[welcome, <ログイン名>] ドロップダウン リストをクリックして、[Change Remote User Role] を選択します。

ステップ 3 [Change Remote User Role] ダイアログボックスで、[User Name]、[Password]、[New Role] の各フィールドに情報を入力し、[Submit] をクリックします。

GUI が更新され、新しいロールが適用されます。

（注） 親ロールに戻るには、もう一度 [Change Remote User Role] ダイアログボックスを開き、[User Name] と [Password] に情報を入力しますが、[New Role] フィールドは空欄のままにしておきます。

REST API を使用したリモート ユーザ ロールの変更

始める前に

ロールは、最初に AVPair と一致するように Cisco ASC サーバーで設定し、その一致に基づいてシェル認証プロファイルとして選択する必要があります。

ユーザーは、ユーザー名 **apicadmin** とパスワードでログインします。

ステップ 1 新しいロールに変更します。

例：

```
<!--   api/requestNewRole/json   -->
<aaaChangeRole>
<attributes userName="apic#tacacs” apicadmin="pwd Ins3965!” role=“newrole"/>
```

ステップ 2 元のロールに戻ります。

例：

```
<!--   api/requestNewRole/json   -->
<aaaChangeRole>
<attributes userName="apic#tacacs” apicadmin="pwd Ins3965!” role=“"/>
```

署名ベースのトランザクションについて

Cisco ACI ファブリックの APIC コントローラは、ユーザを認証するためにさまざまな方法を提供します。

主要な認証方式ではユーザ名とパスワードが使用され、APIC REST API は APIC に対するその後のアクセスに使用できる認証トークンを返します。これは、HTTPS が使用不可であるか有効でない状況では安全でないと見なされます。

提供されている別の認証形式では、トランザクションごとに計算される署名が活用されます。その署名の計算には秘密キーが使用され、そのキーは安全な場所に保管して秘密にしておく必要があります。APIC がトークン以外の署名が付いた要求を受信すると、APIC は X.509 証明書を活用して署名を確認します。署名ベースの認証では、APIC に対するすべてのトランザクションに新しく計算された署名が必要です。これは、ユーザがトランザクションごとに手動で行うタスクではありません。理想的には、この機能は APIC と通信するスクリプトまたはアプリケーションで使用する必要があります。この方法では、攻撃者がユーザクレデンシャルを偽装またはなりすますためには RSA/DSA キーを解読する必要があるため、最も安全です。

（注） また、リプレイ攻撃を防ぐためには HTTPS を使用する必要があります。

認証に X.509 証明書ベースの署名を使用する前に、次の必須タスクが完了していることを確認します。

1. OpenSSL または同様のツールを使用して X.509 証明書と秘密キーを作成します。
2. APIC のローカル ユーザを作成します（ローカル ユーザがすでに利用可能である場合、このタスクはオプションです）。
3. APIC のローカル ユーザに X.509 証明書を追加します。

注意事項と制約事項

次の注意事項と制約事項に従ってください。

- ローカル ユーザはサポートされます。リモート AAA ユーザはサポートされません。

- APIC GUI は証明書認証方式をサポートしません。
- WebSocket と eventchannel は X.509 要求では動作しません。
- サードパーティにより署名された証明書はサポートされません。自己署名証明書を使用します。

アカウンティング

ACI ファブリック アカウンティングは、障害およびイベントと同じメカニズムで処理される以下の 2 つの管理対象オブジェクト（MO）によって処理されます。

- `aaaSessionLR` MO は、APIC およびスイッチでのユーザ アカウントのログイン/ログアウト セッション、およびトークン更新を追跡します。ACI ファブリック セッション アラート機能は、次のような情報を保存します。
  - ユーザ名
  - セッションを開始した IP アドレス
  - タイプ（telnet、https、REST など）
  - セッションの時間と長さ
  - トークン更新：ユーザ アカウントのログイン イベントは、ユーザ アカウントが ACI ファブリックの権利を行使するために必要な、有効なアクティブ トークンを生成します。

（注） トークンはログインに関係なく期限切れになります。ユーザはログアウトできますが、トークンは含まれているタイマー値の期間に従って期限切れになります。

- `aaaModLR` MO は、ユーザがオブジェクトに対して行う変更、およびいつ変更が発生したかを追跡します。
- AAA サーバが ping 可能でない場合は、使用不可としてマークされ、エラーが表示されます。

`aaaSessionLR` と `aaaModLR` の両方のイベント ログが、APIC シャードに保存されます。データがプリセットされているストレージ割り当てサイズを超えると、先入れ先出し方式でレコードを上書きします。

（注） APIC クラスタ ノードを破壊するディスク クラッシュや出火などの破壊的なイベントが発生した場合、イベント ログは失われ、イベント ログはクラスタ全体で複製されません。

aaaModLR MO と aaaSessionLR MO は、クラスまたは識別名（DN）でクエリできます。クラスのクエリは、ファブリック全体のすべてのログ レコードを提供します。ファブリック全体の aaaModLR レコードはすべて、GUI の **[Fabric]** > **[Inventory]** > **[POD]** > **[History]** > **[Audit Log]** セクションから入手できます。APIC GUI の **[History]** > **[Audit Log]** オプションを使用すると、GUI に示された特定のオブジェクトのイベント ログを表示できます。

標準の syslog、callhome、REST クエリ、および CLI エクスポート メカニズムは、aaaModLR MO と aaaSessionLR MO のクエリ データで完全にサポートされます。このデータをエクスポートするデフォルト ポリシーはありません。

APIC には、一連のオブジェクトまたはシステム全体のデータの集約を報告する、事前設定されたクエリはありません。ファブリック管理者は、aaaModLR および aaaSessionLR のクエリ データを定期的に syslog サーバにエクスポートするエクスポート ポリシーを設定できます。エクスポートされたデータを定期的にアーカイブし、システムの一部またはシステム ログ全体のカスタム レポートを生成するために使用できます。

共有サービスとしての外部ネットワークへのルーテッド接続の課金と統計情報

Cisco Application Policy Infrastructure Controller（APIC）は、共有サービスとして外部ネットワークへのルーテッド接続用に設定されたポートからバイト カウントおよびパケット カウント課金統計情報を収集するように設定できます。外部ネットワークは、Cisco Application Centric Infrastructure（ACI）内の外部 L3Out エンドポイント グループ（l3extInstP 管理対象オブジェクト）として表されます。任意のテナントの任意の EPG は、外部ネットワークへのルーテッド接続のために外部 L3Out EPG を共有できます。課金統計情報は、共有サービスとして外部 L3Out EPG を使用するテナントの各 EPG について収集できます。外部 L3Out EPG がプロビジョニングされているリーフスイッチは、課金統計情報を集約先である Cisco APIC に転送します。アカウンティング ポリシーは、これらの課金統計情報を定期的にサーバにエクスポートするように設定できます。

設定

ローカル ユーザの設定

初期の設定スクリプトで、管理者アカウントが設定され、管理者はシステム起動時の唯一のユーザとなります。APIC は、きめ細かなロールベースのアクセス コントロール システムをサポートしており、そのシステムでは、権限が少ない管理者以外のユーザを含め、ユーザアカウントをさまざまなロールで作成することができます。

GUI を使用したローカルユーザーの設定

始める前に

- ACI ファブリックが設置され、APIC コントローラがオンラインになっており、APIC クラスタが形成されて正常に動作していること。

- 必要に応じて、ユーザがアクセスするセキュリティ ドメインが定義されていること。たとえば、新しいユーザ アカウントがテナントへのアクセスに制限される場合、テナント ドメインはそれに応じてタグ付けされます。
- 以下を行うことができる APIC ユーザ アカウントを使用できること。
  - TACACS+ プロバイダーの作成。
  - ターゲット セキュリティ ドメインでのローカル ユーザ アカウントの作成。ターゲット ドメインが `all` である場合、新しいローカル ユーザの作成に使用するログイン アカウントは、`all` にアクセスできるファブリック全体の管理者である必要があります。ターゲット ドメインがテナントである場合、新しいローカル ユーザの作成に使用するログイン アカウントは、ターゲット テナント ドメインに対する完全な読み取り/書き込みアクセス権を持つテナント管理者である必要があります。

ステップ 1 メニュー バーで、**[管理（Admin）] > [AAA]** の順に選択します。

ステップ 2 [Navigation] ペインで、[Users] をクリックします。

作業ペインで、[**ローカル（Local）**] タブが表示されていることを確認します。

ステップ 3 **作業**ペインで、[**アクション（Actions）**]をクリックして、[**ローカルユーザーの作成（Create Local User）**] を選択します。

ステップ 4 [**ユーザー名（Username）**] フィールドにユーザー名を入力します。

ログイン ID は、次のガイドラインを満たしている必要があります。

- APIC 内で一意である必要があります。
- 先頭は英字にする必要があります。
- 1 ー 32 文字を使用できます。
- 英数字、アンダースコア、ハイフンを使用してください。

ユーザーアカウントの作成後は、ユーザー名を変更できません。ユーザアカウントを削除し、新しいユーザ アカウントを作成する必要があります。

ステップ 5 [Password] フィールドにパスワードを入力します。[**確認パスワード（Confirm Password）**] フィールドに同じパスワードを入力します。

ステップ 6 （オプション）ユーザー名の [**説明（Description）**] を入力します。

ステップ 7 [**アカウントのステータス（Account Status）**] オプションを使用して、ユーザーアカウントを有効化または無効化できます。オプションは、アクティブ、非アクティブ、ブロックです。

ステップ 8 ユーザー名に対し、[**姓（Last Name）**]、[**名（First Name）**]、[**電子メールアドレス（Email Address）**]、[**電話番号（Phone Number）**] を入力します。

ステップ 9 セキュリティドメインを追加するには、[**セキュリティドメインの追加（Add Security Domain）**]をクリックします。表示される [**セキュリティドメインの追加（Add Security Domain）**] ウィンドウで、次の詳細を入力します。

a) [セキュリティドメインの選択（Select Security Domain）] をクリックし、ドロップダウンリストからセキュリティドメインを選択します。
b) ロールをユーザー名に関連付けるには、[ロールの選択（Select Role）] をクリックし、ドロップダウンリストからロールを選択します。
c) ドロップダウンリストから [権限タイプ（Privilege Type）] を選択し、チェックマークをクリックして、選択したロールに権限を関連付けます。
d) [追加（Add）] をクリックします。

ステップ 10 [有効期限設定のステータス（Expiration Set Status）] オプションを有効にするには、[有効（Enabled）] のチェックボックスをオンにします。ユーザー名を非アクティブにする日時を設定します。

ステップ 11 [パスワードの更新が必要（Password Update Required）] オプションを有効にするには、[有効（Enabled）] のチェックボックスをオンにします。

ステップ 12 [OTP] オプションを有効にするには、[有効（Enabled）] のチェックボックスをオンにします。

ステップ 13 [ユーザー証明書属性（User Cert Attribute field）] フィールドに、認証証明書からのユーザー ID を入力します。これは、証明書ベースの認証の場合です。

ステップ 14 [X509 証明書（X509 Certificate）] フィールドで、[X509 証明書を追加（Add X509 Certifcate）] をクリックして、名前と証明書の文字列を追加します。

ステップ 15 [SSH 認証（SSH Authorization）] フィールドで、[SSH 認証の追加（Add SSH Authorization）] をクリックして、名前と認証データを追加します。

ステップ 16 [保存（Save）] をクリックします。

GUI を使用した SSH 公開キー認証の設定

始める前に

- ターゲット セキュリティ ドメインでローカル ユーザ アカウントを作成します。ターゲット ドメインが `all` である場合、新しいローカル ユーザの作成に使用するログイン アカウントは、`all` にアクセスできるファブリック全体の管理者である必要があります。ターゲット ドメインがテナントである場合、新しいローカル ユーザの作成に使用するログイン アカウントは、ターゲット テナント ドメインに対する完全な読み取り/書き込みアクセス権を持つテナント管理者である必要があります。
- UNIX コマンド **ssh-keygen** を使用して公開キーを生成します。

  デフォルトのログイン ドメインは **local** に設定する必要があります。

ステップ 1 メニューバーで、[管理者（Admin）] > [ユーザー（Users）] を選択し、[ローカル（Local）] タブが表示されていることを確認します。

ステップ 2 **作業**ペインで、事前に作成したユーザーの名前をクリックします。

ユーザーに関する情報が記載されたウィンドウが右側に表示されます。

ステップ 3 [詳細（Details）] アイコンをクリックすると、新しい画面に  およびユーザーの詳細が表示されます。

下方向にスクロールして SSH 認証の詳細を確認します。

ステップ 4 **[鉛筆（Pencil）]** アイコンをクリックすると、**[ローカルユーザーの編集（Edit Local User）]** 画面が表示されます。必要に応じて、SSH の詳細を変更できます。

（注） リモートロケーションにダウンロードするための SSH 秘密キーファイルを作成するには、メニューバーで、**[ファイル名（Firmware）]** > **[タスクのダウンロード（Download Tasks）]** を展開します。

ステップ 5 **[保存（Save）]** をクリックします。

NX-OS スタイル CLI を使用したローカル ユーザの設定

手順の概要

1. NX-OS CLI で、次に示すようにしてコンフィギュレーション モードを開始します。
2. 新しいユーザを次に示すように作成します。

手順の詳細

ステップ 1 NX-OS CLI で、次に示すようにしてコンフィギュレーション モードを開始します。

例：

```
apic1# configure
apic1(config)#
```

ステップ 2 新しいユーザを次に示すように作成します。

例：

```
apic1(config)# username
 WORD       User name (Max Size 28)
 admin
 cli-user
 jigarshah
 test1
 testUser

apic1(config)# username test
apic1(config-username)#
 account-status      Set The status of the locally-authenticated user account.
 certificate         Create AAA user certificate in X.509 format.
 clear-pwd-history   Clears the password history of a locally-authenticated user
 domain              Create the AAA domain to which the user belongs.
 email               Set The email address of the locally-authenticated user.
 exit                Exit from current mode
 expiration          If expires enabled, Set expiration date of locally-authenticated user account.

 expires             Enable expiry for locally-authenticated user account
 fabric              show fabric related information
 first-name          Set the first name of the locally-authenticated user.
 last-name           Set The last name of the locally-authenticated user.
 no                  Negate a command or set its defaults
```

```
 password             Set The system user password.
 phone                Set The phone number of the locally-authenticated user.
 pwd-lifetime         Set The lifetime of the locally-authenticated user password.
 pwd-strength-check   Enforces the strength of the user password
 show                 Show running system information
 ssh-key              Update ssh key for the user for ssh authentication
 where                show the current mode

apic1(config-username)# exit
```

REST API を使用したローカル ユーザの設定

手順の概要

1. ローカル ユーザを作成します。

手順の詳細

ローカル ユーザを作成します。

例：

```
URL: https://apic-ip-address/api/node/mo/uni/userext.xml
POST CONTENT:
    <aaaUser name="operations" phone="" pwd="<strong_password>" >
        <aaaUserDomain childAction="" descr="" name="all" rn="userdomain-all" status="">
        <aaaUserRole childAction="" descr="" name="Ops" privType="writePriv"/>
        </aaaUserDomain>
    </aaaUser>
```

X.509 証明書と秘密キーの生成

ステップ 1 OpenSSL コマンドを入力して、X.509 証明書と秘密キーを生成します。

例：

```
$ openssl req -new -newkey rsa:1024 -days 36500 -nodes -x509 -keyout userabc.key -out userabc.crt
-subj '/CN=User ABC/O=Cisco Systems/C=US'
```

（注）
- X.509 証明書が生成されると、APIC のユーザ プロファイルに追加され、署名の確認に使用されます。秘密キーは、署名を生成するためにクライアントによって使用されます。
- 証明書には公開キーは含まれていますが、秘密キーは含まれていません。公開キーは、計算された署名を確認するために APIC によって使用される主要な情報です。秘密キーが APIC に保存されることはありません。このキーを秘密にしておく必要があります。

ステップ 2 OpenSSL を使用して証明書のフィールドを表示します。

例：

```
$ openssl x509 -text -in userabc.crt
        Certificate:
            Data:
                Version: 3 (0x2)
                Serial Number:
                    c4:27:6c:4d:69:7c:d2:b6
                Signature Algorithm: sha1WithRSAEncryption
                Issuer: CN=User ABC, O=Cisco Systems, C=US
                Validity
                    Not Before: Jan 12 16:36:14 2015 GMT
                    Not After : Dec 19 16:36:14 2114 GMT
                Subject: CN=User ABC, O=Cisco Systems, C=US
                Subject Public Key Info:
                    Public Key Algorithm: rsaEncryption
                    RSA Public Key: (1024 bit)
                        Modulus (1024 bit):
                            00:92:35:12:cd:2b:78:ef:9d:ca:0e:11:77:77:3a:
                            99:d3:25:42:94:b5:3e:8a:32:55:ce:e9:21:2a:ff:
                            e0:e4:22:58:6d:40:98:b1:0d:42:21:db:cd:44:26:
                            50:77:e5:fa:b6:10:57:d1:ec:95:e9:86:d7:3c:99:
                            ce:c4:7f:61:1d:3c:9e:ae:d8:88:be:80:a0:4a:90:
                            d2:22:e9:1b:25:27:cd:7d:f3:a5:8f:cf:16:a8:e1:
                            3a:3f:68:0b:9c:7c:cb:70:b9:c7:3f:e8:db:85:d8:
                            98:f6:e3:70:4e:47:e2:59:03:49:01:83:8e:50:4a:
                            5f:bc:35:d2:b1:07:be:ec:e1
                        Exponent: 65537 (0x10001)
                X509v3 extensions:
                    X509v3 Subject Key Identifier:
                        0B:E4:11:C7:23:46:10:4F:D1:10:4C:C1:58:C2:1E:18:E8:6D:85:34
                    X509v3 Authority Key Identifier:
                        keyid:0B:E4:11:C7:23:46:10:4F:D1:10:4C:C1:58:C2:1E:18:E8:6D:85:34
                        DirName:/CN=User ABC/O=Cisco Systems/C=US
                        serial:C4:27:6C:4D:69:7C:D2:B6

                    X509v3 Basic Constraints:
                        CA:TRUE
            Signature Algorithm: sha1WithRSAEncryption
                8f:c4:9f:84:06:30:59:0c:d2:8a:09:96:a2:69:3d:cf:ef:79:
                91:ea:cd:ae:80:16:df:16:31:3b:69:89:f7:5a:24:1f:fd:9f:
                d1:d9:b2:02:41:01:b9:e9:8d:da:a8:4c:1e:e5:9b:3e:1d:65:
                84:ff:e8:ad:55:3e:90:a0:a2:fb:3e:3e:ef:c2:11:3d:1b:e6:
                f4:5e:d2:92:e8:24:61:43:59:ec:ea:d2:bb:c9:9a:7a:04:91:
                8e:91:bb:9d:33:d4:28:b5:13:ce:dc:fe:c3:e5:33:97:5d:37:
                cc:5f:ad:af:5a:aa:f4:a3:a8:50:66:7d:f4:fb:78:72:9d:56:
                91:2c
        [snip]
```

REST API を使用したローカル ユーザの作成とユーザ証明書の追加

ローカル ユーザを作成し、ユーザ証明書を追加します。

例：

```
method: POST
        url: http://apic/api/node/mo/uni/userext/user-userabc.json
        payload:
        {
            "aaaUser": {
```

```
                "attributes": {
                    "name": "userabc",
                    "firstName": "Adam",
                    "lastName": "BC",
                    "phone": "408-525-4766",
                    "email": "userabc@cisco.com",
                },
                "children": [{
                    "aaaUserCert": {
                        "attributes": {
                            "name": "userabc.crt",
                          "data": "-----BEGIN CERTIFICATE-----\nMIICjjCCAfegAwIBAgIJAMQnbE <snipped
content> ==\n-----END CERTIFICATE-----",
                        },
                        "children": []
                    },
                    "aaaUserDomain": {
                        "attributes": {
                            "name": "all",
                        },
                        "children": [{
                            "aaaUserRole": {
                                "attributes": {
                                    "name": "aaa",
                                    "privType": "writePriv",
                                },
                                "children": []
                            }
                        }, {
                            "aaaUserRole": {
                                "attributes": {
                                    "name": "access-admin",
                                    "privType": "writePriv",
                                },
                                "children": []
                            }
                        }, {
                            "aaaUserRole": {
                                "attributes": {
                                    "name": "admin",
                                    "privType": "writePriv",
                                },
                                "children": []
                            }
                        }, {
                            "aaaUserRole": {
                                "attributes": {
                                    "name": "fabric-admin",
                                    "privType": "writePriv",
                                },
                                "children": []
                            }
                        }, {
                            "aaaUserRole": {
                                "attributes": {
                                    "name": "nw-svc-admin",
                                    "privType": "writePriv",
                                },
                                "children": []
                            }
                        }, {
                            "aaaUserRole": {
                                "attributes": {
                                    "name": "ops",
```

```
                            "privType": "writePriv",
                        },
                        "children": []
                    }
                }, {
                    "aaaUserRole": {
                        "attributes": {
                            "name": "read-all",
                            "privType": "writePriv",
                        },
                        "children": []
                    }
                }, {
                    "aaaUserRole": {
                        "attributes": {
                            "name": "tenant-admin",
                            "privType": "writePriv",
                        },
                        "children": []
                    }
                }, {
                    "aaaUserRole": {
                        "attributes": {
                            "name": "tenant-ext-admin",
                            "privType": "writePriv",
                        },
                        "children": []
                    }
                }, {
                    "aaaUserRole": {
                        "attributes": {
                            "name": "vmm-admin",
                            "privType": "writePriv",
                        },
                        "children": []
                    }
                }]
            }
        }]
    }
}
```

Python SDK を使用したローカル ユーザの作成

ローカル ユーザを作成します。

例：

```
#!/usr/bin/env python
from cobra.model.pol import Uni as PolUni
from cobra.model.aaa import UserEp as AaaUserEp
from cobra.model.aaa import User as AaaUser
from cobra.model.aaa import UserCert as AaaUserCert
from cobra.model.aaa import UserDomain as AaaUserDomain
from cobra.model.aaa import UserRole as AaaUserRole
from cobra.mit.access import MoDirectory
from cobra.mit.session import LoginSession
```

```
from cobra.internal.codec.jsoncodec import toJSONStr

APIC = 'http://10.10.10.1'
username = 'admin'
password = 'p@$$w0rd'

session = LoginSession(APIC, username, password)
modir = MoDirectory(session)
modir.login()

def readFile(fileName=None, mode="r"):
   if fileName is None:
      return ""
   fileData = ""
   with open(fileName, mode) as aFile:
      fileData = aFile.read()
   return fileData

# Use a dictionary to define the domain and a list of tuples to define
# our aaaUserRoles (roleName, privType)
# This can further be abstracted by doing a query to get the valid
# roles, that is what the GUI does

userRoles = {'all': [
               ('aaa', 'writePriv'),
               ('access-admin', 'writePriv'),
               ('admin', 'writePriv'),
               ('fabric-admin', 'writePriv'),
               ('nw-svc-admin', 'writePriv'),
               ('ops', 'writePriv'),
               ('read-all', 'writePriv'),
               ('tenant-admin', 'writePriv'),
               ('tenant-ext-admin', 'writePriv'),
               ('vmm-admin', 'writePriv'),
           ],
}

uni = PolUni('') # '' is the Dn string for topRoot
aaaUserEp = AaaUserEp(uni)
aaaUser = AaaUser(aaaUserEp, 'userabc', firstName='Adam',
                  email='userabc@cisco.com')

aaaUser.lastName = 'BC'
aaaUser.phone = '555-111-2222'
aaaUserCert = AaaUserCert(aaaUser, 'userabc.crt')
aaaUserCert.data = readFile("/tmp/userabc.crt")
# Now add each aaaUserRole to the aaaUserDomains which are added to the
# aaaUserCert
for domain,roles in userRoles.items():
   aaaUserDomain = AaaUserDomain(aaaUser, domain)
   for roleName, privType in roles:
       aaaUserRole = AaaUserRole(aaaUserDomain, roleName,
                                 privType=privType)
print toJSONStr(aaaUser, prettyPrint=True)

cr = ConfigRequest()
cr.addMo(aaaUser)
modir.commit(cr)
# End of Script to create a user
```

秘密キーを使用した署名の計算

始める前に

次の情報が用意されている必要があります。

- HTTP メソッド：GET、POST、DELETE
- 要求される REST API URI（クエリ オプションを含む）
- POST 要求の場合、APIC に送信される実際のペイロード
- ユーザの X.509 証明書の生成に使用される秘密キー
- APIC のユーザ X.509 証明書の宛先名

ステップ 1 HTTP メソッド、REST API URI、およびペイロードをこの順序で連結し、ファイルに保存します。

OpenSSL で署名を計算するには、この連結データをファイルに保存する必要があります。この例では、ファイル名 payload.txt を使用します。秘密キーは userabc.key というファイルにあることに注意してください。

例：

GET の例：

```
GET http://10.10.10.1/api/class/fvTenant.json?rsp-subtree=children
```

POST の例：

```
POST http://10.10.10.1/api/mo/tn-test.json{"fvTenant": {"attributes": {"status": "deleted", "name":
 "test"}}}
```

ステップ 2 payload.txt ファイルに正しい情報が含まれていることを確認します。

たとえば、前の手順で示したような取得例を使用します。

```
GET http://10.10.10.1/api/class/fvTenant.json?rsp-subtree=children
```

payload.txt ファイルには、次の情報のみ含める必要があります。

```
GET/api/class/fvTenant.json?rsp-subtree=children
```

ステップ 3 payload ファイルを作成するときに新しい行を間違って作成していないことを確認します。

例：

```
# cat -e payload.txt
```

次と同じように出力の最後に **$** 記号があるか確認します。

```
GET/api/class/fvTenant.json?rsp=subtree=children$
```

ある場合、Payload.ファイルを作成したときに新しい行が作成されたことを意味します。payload ファイルの生成時に新しい行が作成されることを防ぐには、次のようなコマンドを使用します。

```
echo -n "GET/api/class/fvTenant.json?rsp-subtree=children" >payload.txt
```

ステップ 4 OpenSSL を使用して、秘密キーとペイロード ファイルを使用して署名を計算します。

例：

```
openssl dgst -sha256 -sign userabc.key payload.txt > payload_sig.bin
```

生成されたファイルには、複数行に印字された署名があります。

ステップ 5 base64 形式に署名を変換します。

例：

```
openssl base64 -A -in payload_sig.bin -out payload_sig.base64
```

ステップ 6 Bash を使用して、署名から改行文字を取り除きます。

例：

```
$ tr -d '\n' < payload_sig.base64
P+OTqK0CeAZjl7+Gute2R1Ww8OGgtzE0wsLlx8fIXXl4V79Zl7
Ou8IdJH9CB4W6CEvdICXqkv3KaQszCIC0+Bn07o3qF//BsIplZmYChD6gCX3f7q
IcjGX+R6HAqGeK7k97cNhXlWEoobFPe/oajtPjOu3tdOjhf/9ujG6Jv6Ro=
```

（注） これは、この特定の要求に関して APIC に送信される署名です。その他の要求では、独自の署名を計算する必要があります。

ステップ 7 署名を文字列内に配置し、APIC が署名をペイロードと照合して確認できるようにします。

この完全な署名が、要求のヘッダー内のクッキーとして APIC に送信されます。

例：

```
APIC-Request-Signature=P+OTqK0CeAZjl7+Gute2R1Ww8OGgtzE0wsLlx8f
IXXl4V79Zl7Ou8IdJH9CB4W6CEvdICXqkv3KaQszCIC0+Bn07o3qF//BsIplZmYChD6gCX3f
7qIcjGX+R6HAqGeK7k97cNhXlWEoobFPe/oajtPjOu3tdOjhf/9ujG6Jv6Ro=;
APIC-Certificate-Algorithm=v1.0; APIC-Certificate-Fingerprint=fingerprint;
APIC-Certificate-DN=uni/userext/user-userabc/usercert-userabc.crt
```

（注） ここで使用される DN が、次のステップの x509 証明書を含むユーザ認定オブジェクトの DN に一致する必要があります。

ステップ 8 署名を使用して APIC と通信するには、Python SDK の CertSession クラスを使用します。

次のスクリプトは、ACI Python SDK の CertSession クラスを使用して、署名を使用して APIC に要求する方法の例です。

例：

```
#!/usr/bin/env python
# It is assumed the user has the X.509 certificate already added to
# their local user configuration on the APIC
from cobra.mit.session import CertSession
from cobra.mit.access import MoDirectory

def readFile(fileName=None, mode="r"):
    if fileName is None:
        return ""
    fileData = ""
    with open(fileName, mode) as aFile:
        fileData = aFile.read()
    return fileData

pkey = readFile("/tmp/userabc.key")
csession = CertSession("https://ApicIPOrHostname/",
                       "uni/userext/user-userabc/usercert-userabc", pkey)
```

```
modir = MoDirectory(csession)
resp = modir.lookupByDn('uni/fabric')
pring resp.dn
# End of script
```

（注） 前のステップで使用した DN が、このステップの x509 証明書を含むユーザ認定オブジェクトの DN に一致する必要があります。

GUI を使用してログイン試行の連続失敗後のユーザー ロックアウトを設定する

ユーザーが設定された回数のログイン試行に失敗した後、そのユーザーをログインできないようにすることができます。特定の期間内にユーザーが何回ログインに失敗可能かを指定できます。ユーザーが何度もログインに失敗すると、そのユーザーは指定された期間ログインできなくなります。

ステップ 1 メニュー バーで、**[管理（Admin）]** > **[AAA]** の順に選択します。

ステップ 2 [Navigation] ペインで、[Security] を選択します。

ステップ 3 [**作業（Work）**] ペインで、[**セキュリティのデフォルト設定（Security Default Settings）**] タブが表示されていることを確認します。

ステップ 4 [鉛筆（pencil）] アイコンをクリックして、次のフィールドを編集します。

a) **ログインに複数回失敗した後でユーザーをロックアウトする**には、**[有効化（Enable）]** を選択します。

b) **[ユーザーがロックアウトされるまでの試行失敗回数（Number of failed attempts before user is locked out）]** に、目的の値を入力します。

有効な範囲は 1 ～ 15 です。デフォルトは 5 分です。

c) **[連続して試行が失敗した期間 (m)（Time period in which consecutive attempts were failed (m)）]** に、Cisco Application Policy Infrastructure Controller（APIC）が失敗した試行をカウントする時間間隔の値を分単位で入力します。

範囲は 1 ～ 720 時間です。デフォルトは 5 分です。

d) **[ロックアウトの持続時間 (m)（Duration of lockout (m)）]** には、ユーザーが何度もログインに失敗したことを理由ににロックアウトされる時間を分単位で入力します。

ステップ 5 [送信（Submit）] をクリックします。

OTP ベース認証向けローカルユーザーの設定

次の手順では、Cisco APIC GUI を使用してローカル ユーザーの OTP ベースの 2 要素認証を構成します。この手順では、ファブリック管理者を想定しています。

始める前に

OTP ベースの 2 要素認証を有効にするローカル ユーザーをすでに作成している必要があります。

ステップ 1 メニュー バーで、**[管理（Admin）]** > **[AAA]** の順に選択します。

ステップ 2 **[ナビゲーション（Navigation）]** ペインで **[ユーザー（Users）]** を選択します。

ステップ 3 **作業**ペインで、OTP ベース二要素認証を有効にするユーザーをクリックします。

ユーザーに関する詳細が記載されたウィンドウが右側に表示されます。**[詳細（Details）**（  ）] アイコンをクリックして、**[鉛筆（Pencil）**（編集（Edit））]アイコンをクリックするか、**[アクション（Actions）]** アイコン > **[編集（Edit）]** の順にクリックします。

ステップ 4 下にスクロールし、[詳細設定（Advanced Settings）] で、**[OTP を有効にする（Enable OTP）]** チェックボックスをオンにします。

ステップ 5 [保存（Save）] をクリックします。

ステップ 6 **[ポリシーの使用状況の警告（Policy Usage Warning）]** ダイアログで **[変更を送信（Submit Changes）]** をクリックします。

この時点から、電子メールなどを通じてユーザーに QR コードまたは OTP キーを提供するか、ユーザーが Cisco APIC GUI にログインしてコードとキーを直接取得することができます。

ステップ 7 （任意）ユーザーに QR コードまたは OTP キーを提供する場合は、サブステップに進みます。それ以外の場合は、ここで停止します。

a) 再度ユーザーをダブルクリックします。

b) **OTP キー**の右側にある展開ボタンをクリックします。

QR コードと OTP キーを含むダイアログが表示されます。

c) ユーザーに QR コードまたは OTP キーを提供します。

d) Cisco APIC で、**[閉じる（Close）]** をクリックします。

e) [閉じる（Close）] をクリックします。

次のタスク

OTP を有効にしたユーザーは、OTP 認証の構成を完了する必要があります。「GUI を使用してユーザーによる OTP ベース 2 要素認証の設定を完了する （34 ページ）」を参照してください。

GUI を使用してユーザーによる OTP ベース 2 要素認証の設定を完了する

次の手順では、Cisco APIC GUI を使用した OTP ベースの 2 要素認証の設定を完了します。この手順は、ファブリック管理者が OTP ベースの 2 要素認証を有効にしたユーザーであることを前提としています。

始める前に

ファブリック管理者は、アカウントに対して OTP ベースの 2 要素認証を有効にしている必要があります。

ステップ 1 Android または Apple iOS スマートフォンで、適切な 2 要素認証アプリをダウンロードします。

ステップ 2 ファブリック管理者から、または Cisco APIC GUI にログインして、QR コードまたは OTP キーを取得します。

GUI にログインすると、資格情報を入力すると、QR コードと OTP キーが表示されます。

ステップ 3 スマートフォンを使用して QR コードをスキャンし、2 要素認証アプリの指示に従うか、Cisco APIC GUI で OTP キーを入力します。