Cisco Finesse(12.6 ES03)へのVPNレスアクセスのためのNginxリバースプロキシの設定

内容

はじめに

このドキュメントでは、12.6 ES03バージョンのCisco Finesse、Cisco Unified Intelligence Center(CUIC)、およびCisco Identity Service(IdS)に基づいてVPNに接続せずに、逆プロキシを使用してCisco Finesseデスクトップにアクセスする方法について説明します。

注:Nginxのインストールと設定は、シスコではサポートしていません。このテーマに関する質問については、シスココミュニティフォーラムで議論できます。

注：VPN-LessのES03導入の場合は、個々のコンポーネントのreadmeを参照して、アップグレードを計画し、互換性の制限を確認してください。Cisco Finesse 12.6 ES03 Readme、CUIC / IdS 12.6 ES03 Readme

前提条件

要件

次の項目に関する知識があることが推奨されます。

- Cisco Unified Contact Center Enterprise(UCCE)リリース
- Cisco Finesse
- Linuxの管理
- ネットワーク管理とLinuxネットワーク管理

使用するコンポーネント

このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づいています。

- Finesse:12.6 ES03
- CUIC:12.6 ES03
- IdS:12.6 ES03
- UCCE/Hosted Collaboration Solution(HCS)for Contact Center(CC) - 11.6以降
- Packaged Contact Center Enterprise(PCCE)- 12.5以降

  注：LD/CUICの共存による導入のため、PCCE/UCCE 2kの導入はCCE 12.6バージョン上で行う必要があります。

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

注：このドキュメントで示す設定は、CentOS 8.0に導入されたNginxリバースプロキシ(OpenResty)を使用して、2000ユーザのUCCE導入サンプルと比較して設定、強化、およびロードテストされています。パフォーマンスプロファイルのリファレンス情報は、このドキュメントで提供されています。

背景説明

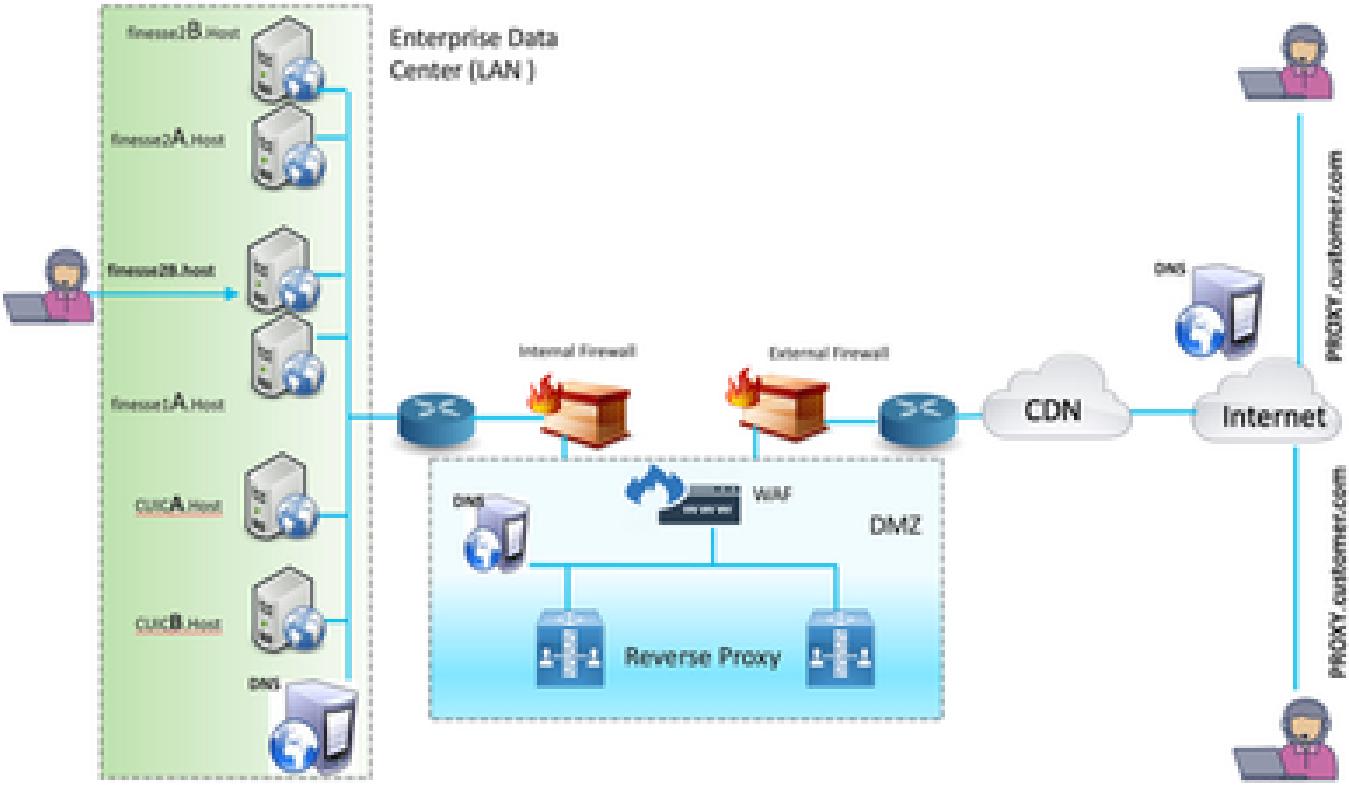
この導入モデルは、UCCE/PCCEおよびUCCEソリューション向けHCSでサポートされています。

VPNに接続せずにCisco Finesseデスクトップにアクセスするオプションとして、逆プロキシの導入がサポートされています（12.6 ES01から使用可能）。この機能により、エージェントはインターネットを介してどこからでもFinesseデスクトップにアクセスできる柔軟性が得られます。

この機能を有効にするには、逆プロキシペアを非武装地帯(DMZ)に導入する必要があります。

逆プロキシの導入では、メディアアクセスは変更されません。エージェントがメディアに接続するには、Cisco Jabber over Mobile and Remote Access(MRA)ソリューションを使用するか、公衆電話交換網(PSTN)またはモバイルエンドポイントを使用するUCCEのモバイルエージェント機能を使用します。この図は、リバースプロキシノードの単一のハイアベイラビリティ(HA)ペアを介して2つのFinesseクラスタと2つのCUICノードにアクセスする場合のネットワーク展開の様子を示しています。

インターネット上のエージェントとLANから接続するエージェントからの同時アクセスは、次の図に示すようにサポートされます。



注：この導入をサポートするには、Nginxの代わりにサードパーティプロキシ選択基準の機能ガイドを参照してください。

- UCCE 12.6機能ガイド:VPNレス機能の概要、設計、および設定の詳細について説明します。
- UCCE 12.6セキュリティガイド：逆プロキシ導入のセキュリティ設定のガイドラインを提供します。

このドキュメントを読む前に、機能ガイドおよびセキュリティガイドの「VPNレス」セクションを参照することをお勧めします。

ES03の変更点

- 新しい機能
  - Finesseスーパーバイザ機能は、リバースプロキシを介してサポートされるようになりました。
  - CUICのリアルタイムおよび履歴レポートが、プロキシ環境のFinesseガジェットでサポートされるようになりました。
  - すべての要求/通信の認証：Luaのサポートが必要
    - すべてのFinesse/CUIC/IM & Presence(IM&P)要求は、データセンターに入る前にプロキシで認証されます。
    - Websocketおよびライブデータソケットio接続も制限され、Finesseに対してセキュリティ保護された要求を正常に行ったクライアントからのみ許可されます。

    - Fail2Banと併用して悪意のあるIPアドレスをブロックできる、プロキシでの総当たり攻撃の検出とロギング。

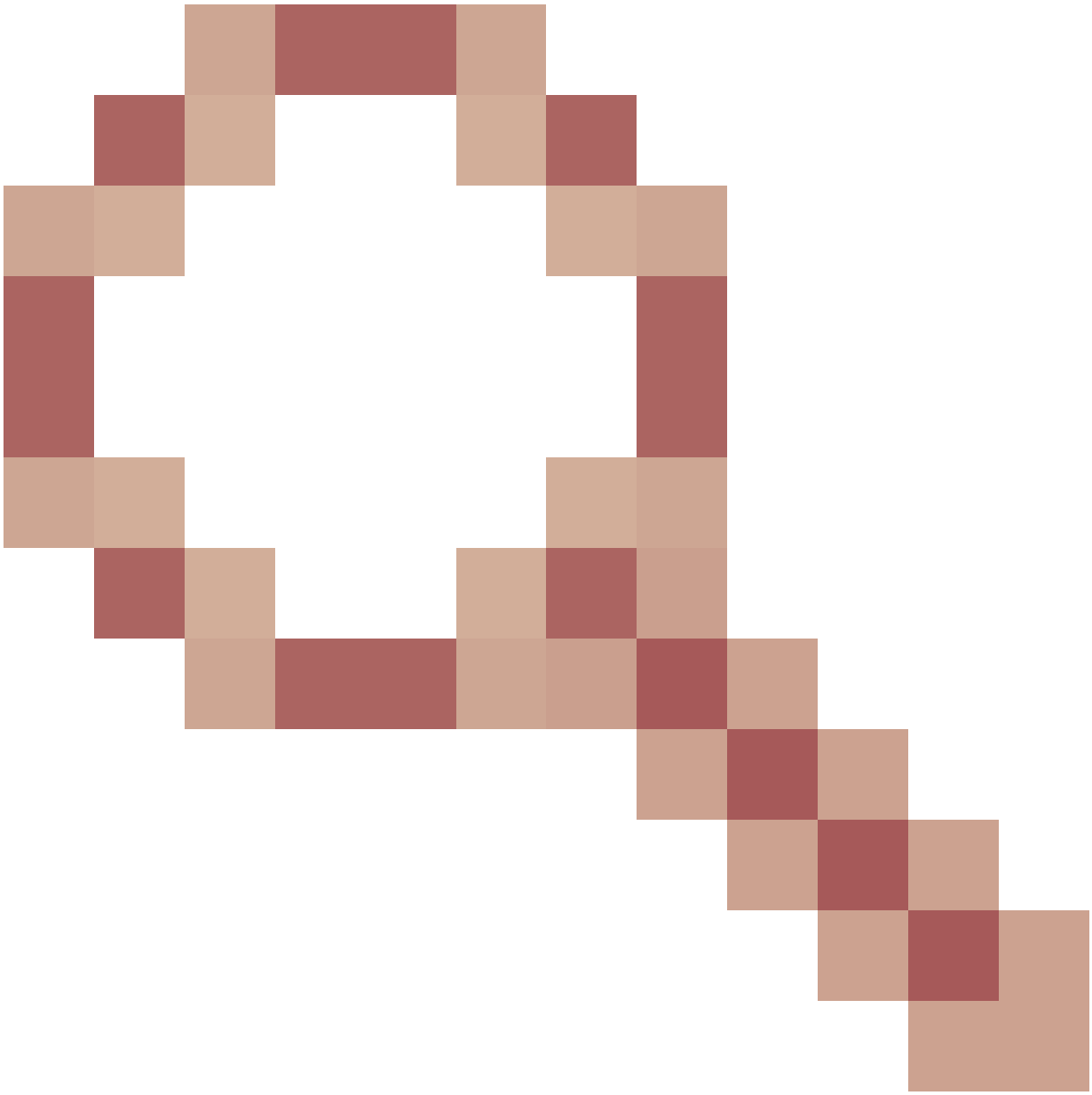
- リバースプロキシ設定のセキュリティ拡張：Luaのサポートが必要
    - 逆プロキシコンポーネントとアップストリームコンポーネント(Finesse/IdS/CUIC/Livedata)間の相互Transport Layer Security(TLS)認証
    - SeLinuxの設定
    - プロキシおよびコンポーネントサーバーの要求に対する相互SSL (Secure Sockets Layer)信頼検証を有効にします。

- サービス拒否(DoS)/分散型サービス拒否(DDoS)攻撃を防ぐためのプロキシ設定のセキュリティ強化 – Luaのサポートが必要
    - システムのさまざまな部分に対する拡張Nginx要求レート制限。
    - IpTableのレート制限。
    - アップストリームコンポーネントサーバを要求する前の静的リソース要求の検証。
    - アップストリームのコンポーネントサーバにヒットしない、より軽量でキャッシュ可能な非認証ページ。

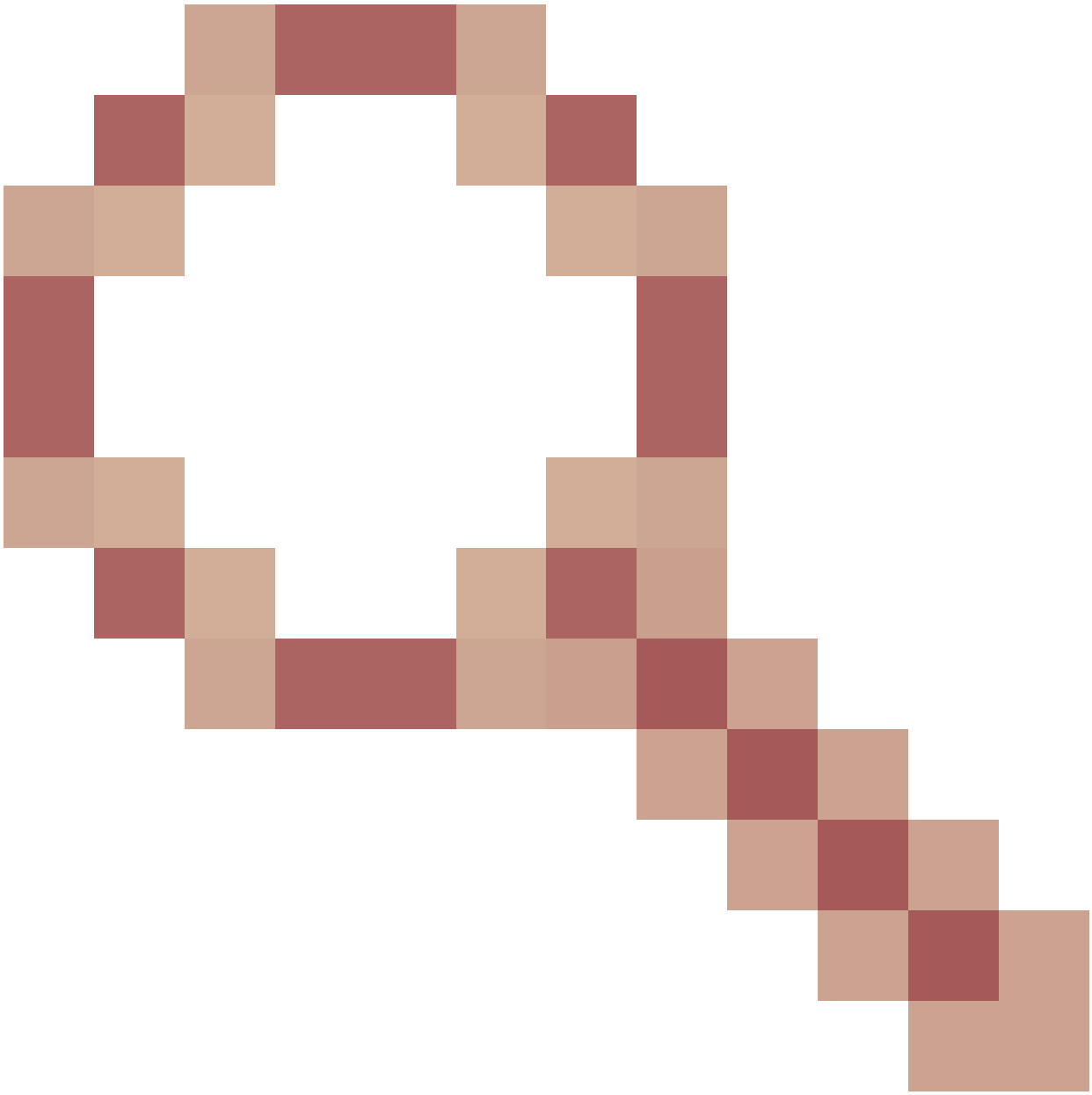
- その他の機能：Luaのサポートが必要
    - 自動設定を支援し、パフォーマンスを向上させるために、プロキシから提供されるAuto Sensing Cross-Origin Resource Sharing(CORS)応答

- VPNレスに関連する不具合修正

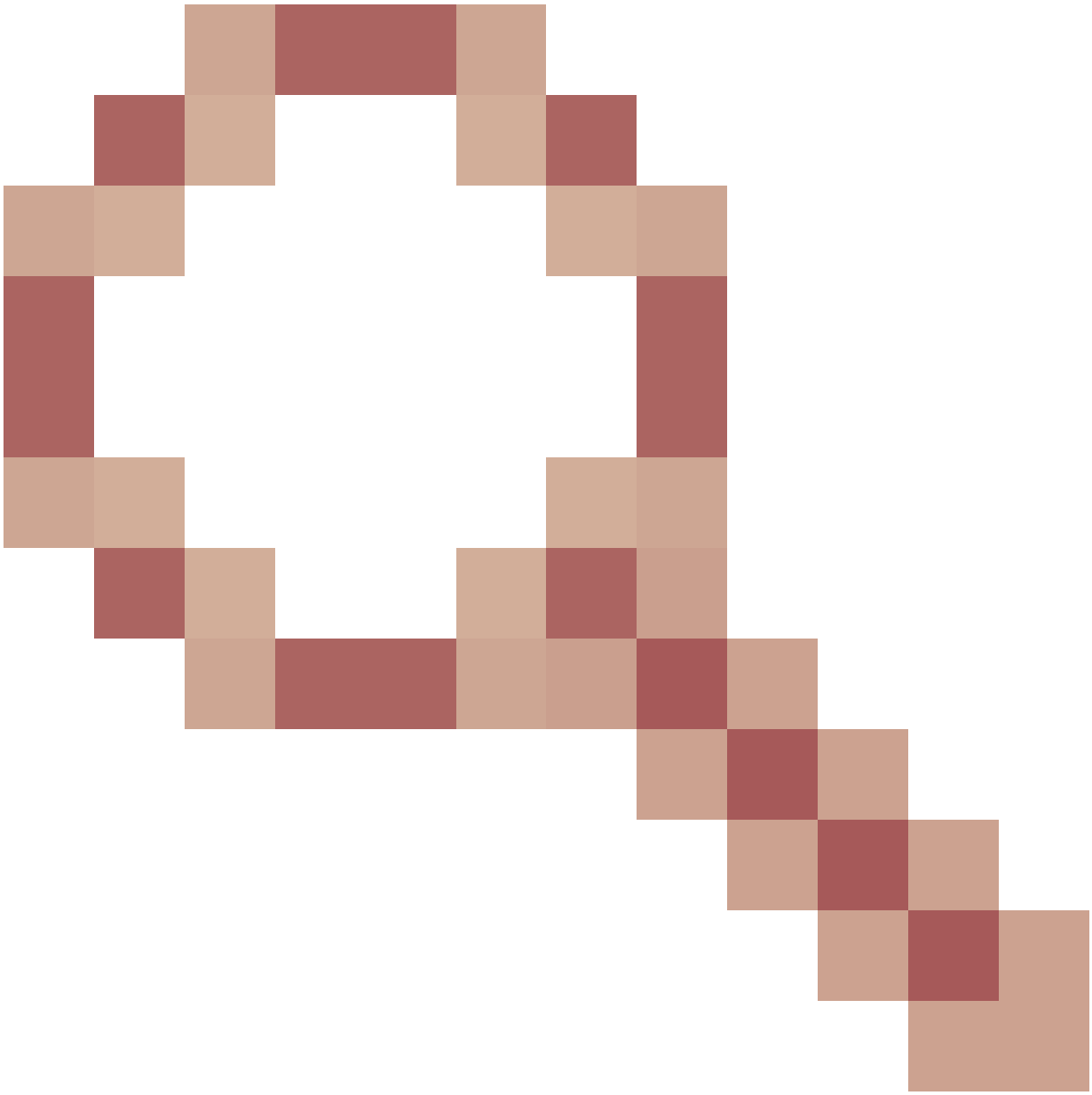
    - CSCwa26057



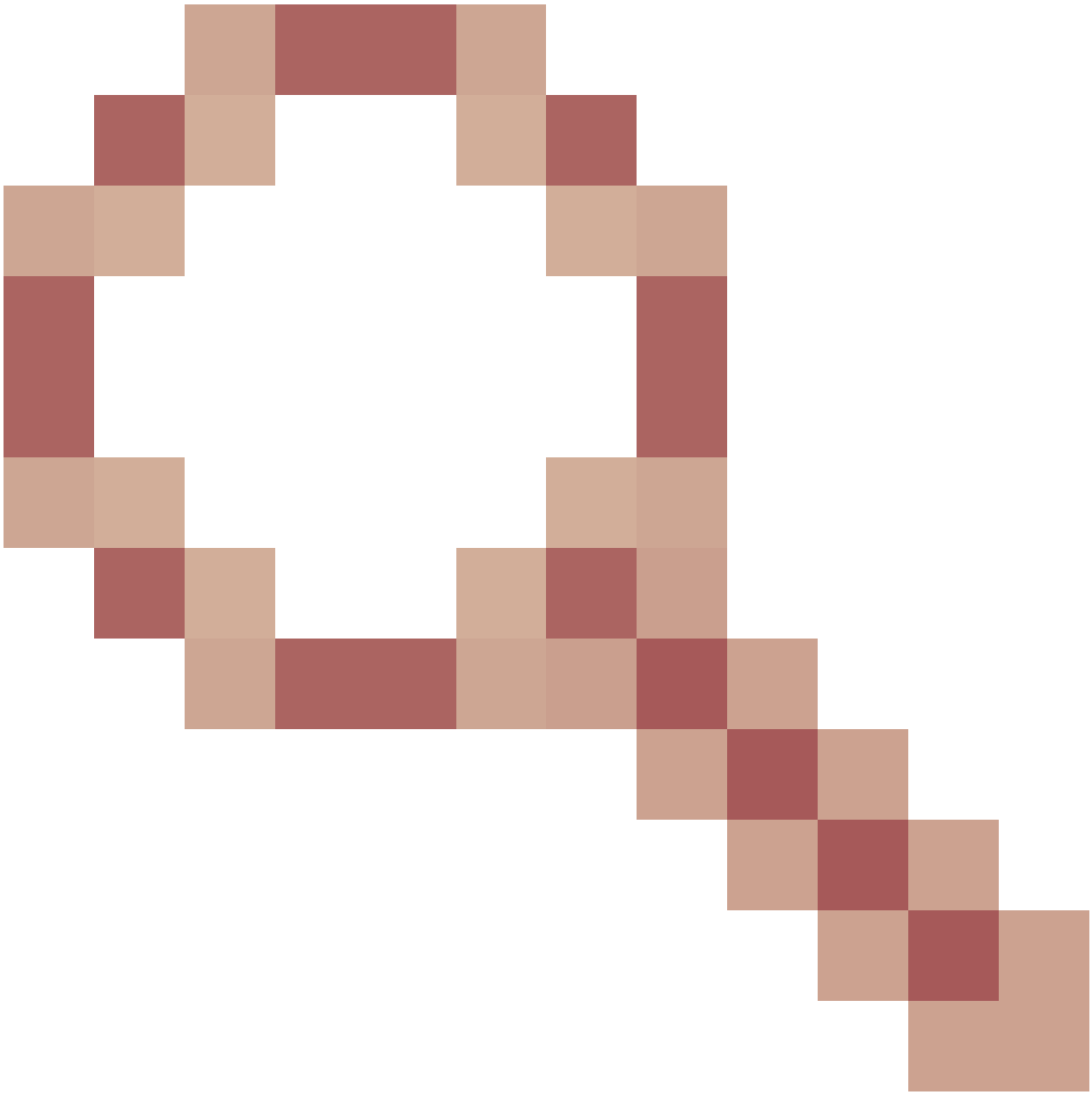
CSCwa26057



/>- Finesseデスクトップログイン中にエージェントに提供される複数の証明書
- CSCwa24471

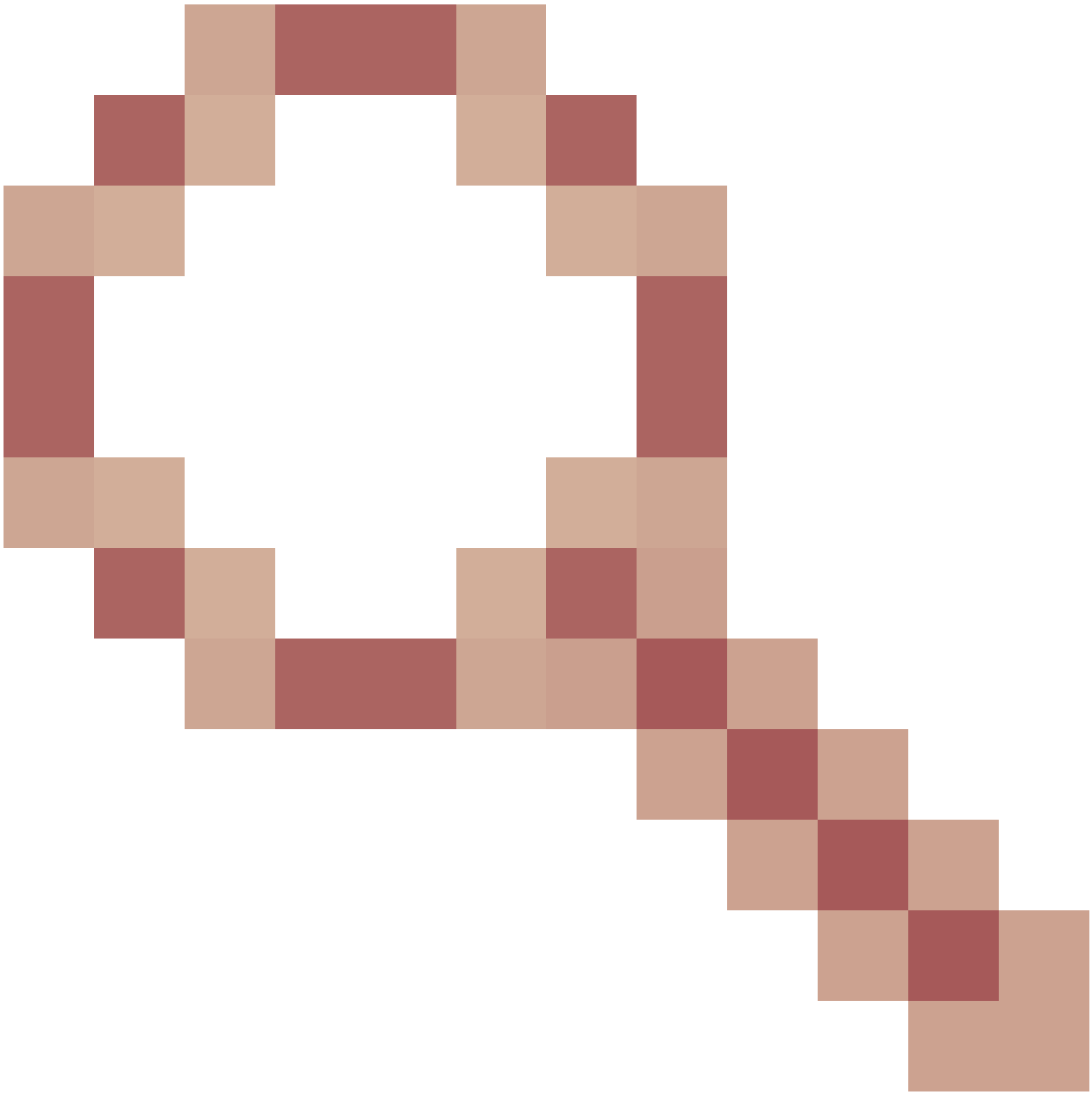


- Finesse口グインページにSSOエージェントのFQDN名が表示されない
  - CSCwa24519



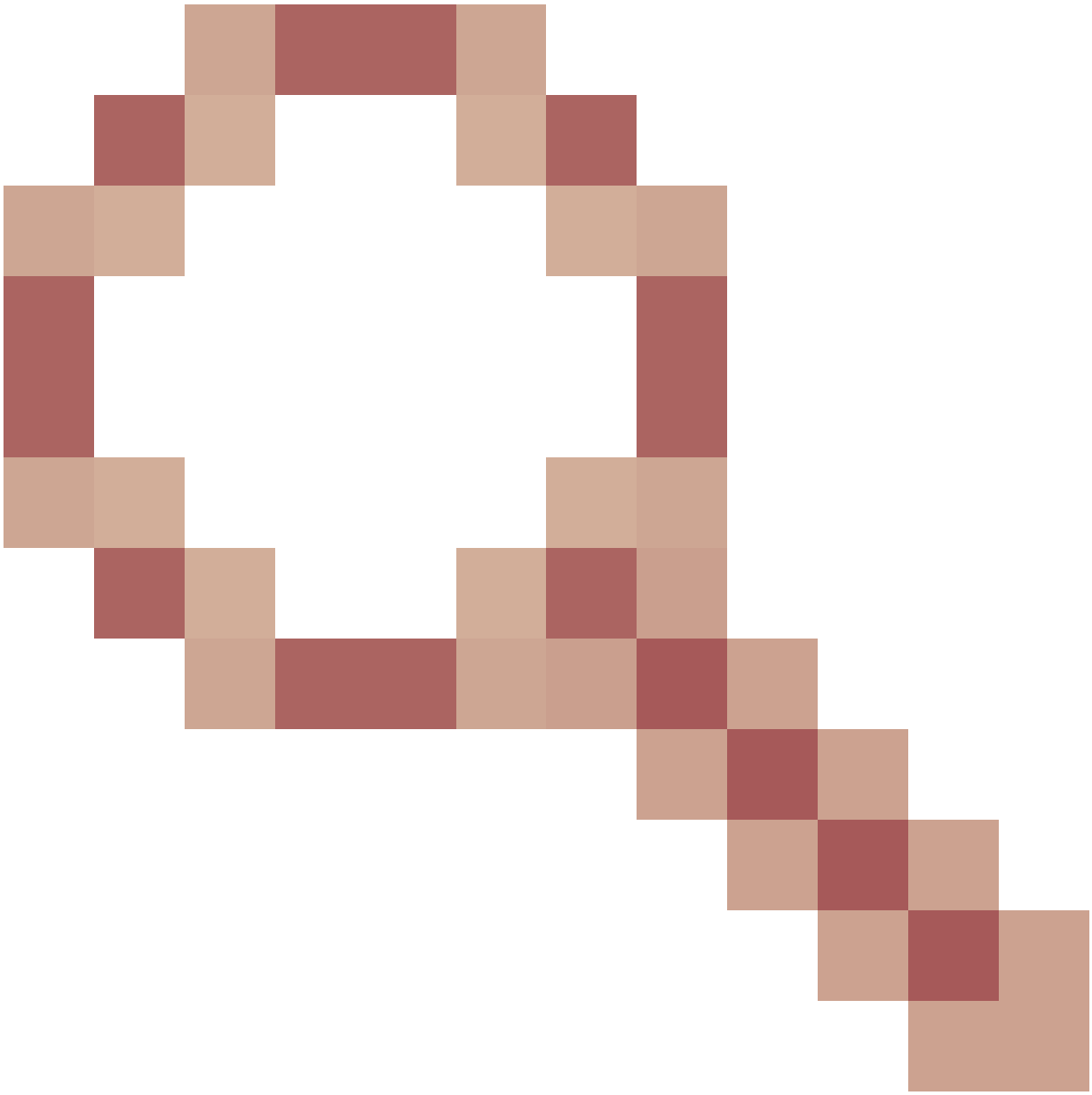
：コンポーネントから逆プロキシホスト名を解決できない場合、Webプロキシサービスを再起動できない

- CSCwa23252



：プロキシFinesseの信頼は、深さがCA証明書チェーンに対して複数の場合に壊れます

- CSCwa46459



webサービスに公開されるlog4jのゼロデイ脆弱性

ES01ベースのVPNレス構成のアップグレードノート

- ES03設定では、NginxのインストールとLuaのサポートが必要です。
- 証明書の要件
  - Cisco Finesse、CUIC、およびIdSでは、Nginx ES02設定がアップストリームサーバに正常に接続できるようになる前に、Nginx/OpenRestyホスト証明書をTomcat信頼ストアに追加して再起動する必要があります。
  - ES03ベースの設定を使用するには、Cisco Finesse、CUIC、およびIdSアップストリームサーバ証明書をNginxサーバで設定する必要があります。

注:ES03 Nginx設定をインストールする前に、既存のES01ベースのNginx設定を削除することをお勧めします。

注:ES03設定スクリプトでは、Cisco Finesse、CUIC、およびIdSに対応するES03 COPをイ

ンストールする必要もあります。

[Authentication]

Finesse 12.6 ES03では、プロキシでの認証が導入されています。認証は、シングルサインオン(SSO)および非SSO展開でサポートされます。

認証は、アップストリームのコンポーネントサーバに転送される前に、プロキシで受け入れられるすべての要求とプロトコルに対して適用されます。アップストリームのコンポーネントサーバでは、コンポーネントサーバによってローカルに適用される認証も実行されます。すべての認証では、共通のFinesseログインクレデンシャルを使用して要求を認証します。

認証や接続後にExtensible Messaging and Presence Protocol(XMPP)などのアプリケーションプロトコルに依存するWebソケットなどの固定接続は、ソケット接続を確立する前にアプリケーション認証が成功した元のIPアドレスを検証することで、プロキシで認証されます。

非SSO認証

非SSO認証では追加の設定は必要なく、必要なスクリプトの置き換えが行われると、すぐにNginx設定スクリプトを使用できます。認証は、Finesseへのログインに使用されるユーザ名とパスワードに依存します。すべてのエンドポイントへのアクセスは、Finesse認証サービスによって検証されます。

有効なユーザのリストは、プロキシでローカルにキャッシュされます（キャッシュは15分ごとに更新されます）。これは、要求でユーザを検証するために使用されます。ユーザクレデンシャルは、設定されたFinesse URIに要求を転送することによって検証されます。その後、クレデンシャルハッシュがローカルにキャッシュされ（15分間キャッシュ）、新しい要求がローカルに認証されます。ユーザ名またはパスワードに変更があった場合は、15分後に有効になります。

SSO認証

SSO認証では、管理者がコンフィギュレーションファイル内のNginxサーバでIdSトークン暗号化キーを設定する必要があります。IdSトークンの暗号キーは、IdSサーバからshow ids secret CLIコマンドを使用して取得できます。これらのキーは、SSO認証が機能する前に、管理者がスクリプトで実行する必要がある#Must-change置換の1つとして設定する必要があります。

IdSでプロキシ解決を機能させるために実行するIdS SAML設定については、SSOユーザガイドを参照してください。

SSO認証を設定すると、有効なトークンのペアを使用して、システム内の任意のエンドポイントにアクセスできるようになります。プロキシ設定は、IdSに対するトークン取得要求を代行受信するか、有効なトークンを復号化し、それ以降の検証のためにローカルにキャッシュすることによって、クレデンシャルを検証します。

Websocket接続の認証

カスタムヘッダーはブラウザのネイティブwebsocket実装でサポートされていないため、

WebSocket接続は標準の認証ヘッダーでは認証できません。アプリケーションレベルの認証プロトコル。ペイロードに含まれる認証情報によってWebSocket接続の確立が阻止されないため、悪意のあるエンティティが、無数の接続を作成してシステムを圧倒するだけでDOSまたはDDOS攻撃を仕掛ける可能性があります。

この可能性を軽減するために、提供されているnginx reverse proxy設定には、websocket接続の確立前に認証されたREST要求を正常に実行したIPアドレスからのwebsocket接続のみを許可するための特定のチェックがあります。つまり、REST要求が発行される前にWebSocket接続の作成を試行するクライアントは、認証に失敗したことを示すエラーを受け取るようになります。これはサポートされている使用シナリオではありません。

総当たり攻撃の防止

Finesse 12.6 ES02認証スクリプトは、ユーザパスワードの推測に使用できるブルートフォースアタックをアクティブに防止します。これは、サービスへのアクセスに使用されるIPアドレスを、一定回数短時間で失敗した後にブロックすることで行われます。これらの要求は、418 client errorによって拒否されます。ブロックされたIPアドレスの詳細には、ファイル<nginx-install-directory>/logs/blocking.logおよび<nginx-install-directory>/logs/error.logからアクセスできます。

失敗した要求の数、時間間隔、およびブロック期間は設定可能です。設定は、<nginx-install-directory>/conf/conf.d/maps.confファイルにあります。

```
## These two constants indicate five auth failures from a client can be allowed in thirty seconds.
## if  the threshold is crossed,client ip will be blocked.
map $host $auth_failure_threshold_for_lock {
    ## Must-change Replace below two parameters as per requirement
    default 5 ;
}

map $host $auth_failure_counting_window_secs {
    ## Must-change Replace below two parameters as per requirement
    default 30;
}

## This indicates duration of blocking a client to avoid brute force attack
map $host $ip_blocking_duration {
    ## Must-change Replace below parameter as per requirement
    default 1800;
}
```

Logging

ブロックされているIPアドレスを見つけるには、ディレクトリ<nginx-install-directory>/logsから次のコマンドを実行します。

```
grep "will be blocked for" blocking.log
grep "IP is already blocked." error.log
```

```
2021/10/29 17:30:59 [emerg] 1181750#1181750: *19 [lua] block_unauthorized_users.lua:153:
_redirectAndSendError(): 10.68.218.190 will be blocked for 30 minutes for exceeding retry limit.,
client: 10.68.218.190, server: saproxy.cisco.com, request:
"GET /finesse/api/SystemInfo?nocache=1636456574482 HTTP/2.0", host: "saproxy.cisco.com:8445",
referrer: "https://saproxy.cisco.com:8445/desktop/container/?locale=en_US&"

2021/10/29 19:21:00 [error] 943068#943068: *43 [lua] block_unauthorized_users.lua:53: 10.70.235.30 ::
IP is already blocked..., client: 10.70.235.30, server: saproxy.cisco.com, request:
"GET /finesse/api/SystemInfo?nocache=1635591686497 HTTP/2.0", host: "saproxy.cisco.com:8445",
referrer: "https://saproxy.cisco.com:8445/desktop/container/?locale=en_US"
```

Fail2banまたはIPtable/firewallルールに禁止を追加する同様のツールと統合することを推奨します。

Fail2banのインストールと設定

Fail2banはログファイルをスキャンし、悪意のある兆候を示すIPを禁止します。パスワードの失敗が多すぎる、不正利用を探すなど。一般に、任意の他のアクション（電子メールの送信など）も設定できますが、指定された期間IPアドレスを拒否するようにファイアウォールルールを更新するためにFail2Banが使用されます。詳細については、https://www.fail2ban.org/を参照してください。

Fail2banは、blocking.logを監視するように設定できます。これにより、Bruteforce攻撃を検出した際にNginxによってブロックされたIPアドレスを特定し、設定可能な期間だけブロックすることができます。CentOSリバースプロキシにfail2banをインストールして設定する手順は、次のとおりです。

1. yumを使用してFail2banをインストールします。

```
yum update && yum install epel-release
yum install fail2ban
```

2. ローカルの刑務所を作成します。

Jail構成では、管理者は、ブロックされたIPアドレスによるアクセスを禁止するポート、ブロックされた状態のIPアドレスの期間、監視するログ・ファイルからブロックされたIPアドレスを識別するために使用するフィルタ構成など、さまざまなプロパティを構成できます。アップストリームサーバへのアクセスをブロックするIPアドレスを禁止するカスタム設定を追加する手順は、次のとおりです。

2.1. Fail2banインストールディレクトリに移動します（この例では/etc/fail2ban）

```
cd /etc/fail2ban
```

2.2. jail.confのコピーをjail.localに作成し、ローカルでの変更を分離します。

```
cp jail.conf jail.local
```

2.3.これらのjail構成をjail.localファイルの最後に追加し、テンプレート内のポートを実際のポートに置き換えます。必要に応じて禁止時間設定を更新します。

```
# Jail configurations for HTTP connections.
[finesse-http-auth]
enabled = true
# The ports to be blocked. Add any additional ports.
port = http,https,<finesse-ports>,<cuic-ports>,<any-other-ports-to-be-blocked>
# Path to nginx blocking logs.
logpath = /usr/local/openresty/nginx/logs/blocking.log
# The filter configuration.
filter = finesseban
# Block the IP from accessing the port, once the IP is blocked by lua.
maxretry= 1
# Duration for retry set to 3 mins. Doesn't count as the maxretry is 1
findtime= 180
# Lock time is set to 3 mins. Change as per requirements.
bantime = 180
```

3. フィルタを構成します。

フィルタはFail2banに対し、禁止するホストを特定するために口グで何を探すかを指示します。フィルタを作成する手順は、次のとおりです。

3.1. filter.d/finesseban.confを作成します。

```
touch filter.d/finesseban.conf
```

3.2.これらの行をfilter.d/finesseban.confファイルに追加します。

```
[Definition]
# The regex match that would cause blocking of the host.
failregex = <HOST> will be blocked for
```

4. Fail2banを起動します。

fail2banを起動するには、次のコマンドを実行します。

```
fail2ban-client start
```

fail2banログファイルを開き、エラーがないことを確認します。デフォルトでは、fail2banのログはファイル/var/log/fail2ban.logに記録されます。

静的リソースURLの検証

認証なしでアクセスできる有効なエンドポイントはすべて、ES03スクリプトでアクティブに追跡されます。

無効なURIが要求された場合、これらの非認証パスに対する要求は、アップストリームサーバに送信されることなく、アクティブに拒否されます。

CORSヘッダーのキャッシュ

最初のオプション要求が成功すると、応答ヘッダーaccess-control-allow-headers、access-control-allow-origin、access-control-allow-methods、access-control-expose-headers、およびaccess-control-allow-credentialsが5分間プロキシでキャッシュされます。これらのヘッダーは、それぞれのアップストリームサーバに対してキャッシュされます。

設定

このドキュメントでは、Finesse VPNレスアクセスを有効にするために使用する逆プロキシとしてNginxを設定する方法について説明します。提供された手順を検証するために使用されるUCCEソリューションコンポーネント、プロキシ、およびOSバージョンが示されています。関連する手順は、選択したOS/プロキシに合わせて調整する必要があります。

- 使用するNginxバージョン：OpenResty 1.19.9.1
- 設定に使用するOS:CentOS 8.0

注：説明されているNginxの設定は、Finesse Release 12.6(1)ES3ソフトウェアダウンロードページからダウンロードできます。

VPNレスアクセス用のソリューションコンポーネントの設定

プロキシの設定後、ソリューションコンポーネント(Finesse/CUIC/IdS)をVPNレスアクセス用に設定します。次のコマンドを使用して、ソリューションへのアクセスに使用するプロキシ/サービスのホスト名とIPを計画します。

```
utils system reverse-proxy allowed-hosts add
utils system reverse-proxy config-uri <uri> add
```

これらのコマンドの詳細は「UCCE 12.6機能ガイド」に記載されており、このドキュメントを使用する前に参照する必要があります。

DMZでの逆プロキシとしてのOpenRestyのインストール

このセクションでは、OpenRestyベースのプロキシインストール手順について詳しく説明します。リバースプロキシは、通常、前述の導入図に示すように、ネットワーク非武装地帯(DMZ)の専用デバイスとして設定されます。

1. 必要なハードウェア仕様を使用して、任意のOSをインストールします。カーネルとIPv4パラメータの微調整は、選択したOSによって異なる場合があります。選択したOSのバージョンが異なる場合は、これらの側面を再確認することをお勧めします。
2. 2つのネットワークインターフェイスを設定します。1つのインターフェイスは、インターネットクライアントからのパブリックアクセスに必要で、別のインターフェイスは内部ネットワーク内のサーバと通信するために必要です。
3. OpenRestyをインストールします。

Nginx 1.19+をベースとし、Luaをサポートしている限り、この目的には任意の種類のNginxを使用できます。

- Nginxプラス
- Nginxオープンソース（使用するには、NginxオープンソースをOpenRestyベースのLuaモジュールとともにコンパイルする必要があります）
- OpenResty
- GetPageSpeedエキストラ

注：提供されている設定はOpenResty 1.19でテスト済みであり、他のディストリビューションでも機能することが期待できます。ただし、マイナーアップデートが行われた場合に限ります。

OpenRestyのインストール

1. OpenRestyをインストールします。「OpenResty Linuxパッケージ」を参照してください。OpenRestyのインストールの一環として、Nginxがこの場所にインストールされ、~/.bashrcファイルにOpenRestyパスを追加してPATH変数に追加されます。

   ```
   export PATH=/usr/local/openresty/bin:$PATH
   ```

2. Nginxを開始/停止します。
   - Nginxを起動するには、**openresty**を入力します。
   - Nginxを停止するには、**openresty -s stop**を入力します。

Nginxの設定

OpenRestyベースのNginxインストールの設定について説明します。OpenRestyのデフォルトディ

レクトリは次のとおりです。

- <nginx-install-directory> = /usr/local/openresty/nginx
- <Openresty-install-directory> = /usr/local/openresty

1. Nginxのリバースプロキシ設定を含むFinesseリリース12.6(1)ES03ソフトウェアダウンロードページ(12.6-ES03-reverse-proxy-config.zip)からファイルをダウンロードして抽出します。
2. 抽出した逆プロキシ設定ディレクトリから、<nginx-install-directory>/conf、<nginx-install-directory>/conf/conf.d/、および<nginx-install-directory>/html/にそれぞれ、nginx.conf、nginx/conf.d/、およびnginx/html/をコピーします。
3. 抽出した逆プロキシ設定ディレクトリからnginx/luaディレクトリを<nginx-install-directory>内にコピーします。
4. lualibの内容を<Openresty-install-directory>/lualib/restyにコピーします。
5. nginx/logrotate/saproxyファイルを<nginx-install-directory>/logrotate/フォルダにコピーして、nginxログローテーションを設定します。Nginxのデフォルトが使用されていない場合は、正しいログディレクトリを指すようにファイルの内容を変更します。
6. Nginxは、専用の非特権サービスアカウントを使用して実行する必要があります。このアカウントはロックされ、無効なシェルが設定されている必要があります（または、選択したOSに該当します）。
7. htmlおよびconf.dという名前の抽出されたフォルダの下にあるファイルで「Must-change」という文字列を見つけ、表示された値を適切なエントリで置き換えます。
8. コンフィギュレーションファイルのMust-changeコメントで説明されている、すべての必須の交換が行われていることを確認します。
9. CUICおよびFinesse用に設定されたキャッシュディレクトリが、これらの一時ディレクトリとともに<nginx-install-directory>/cacheの下に作成されていることを確認します。
    - <nginx-install-directory>/cache/client_temp
    - <nginx-install-directory>/cache/proxy_temp

注：ここに示す設定は2000年導入の例であり、より大規模な導入に合わせて適切に拡張する必要があります。

Nginxキャッシュの設定

デフォルトでは、プロキシキャッシュパスはファイルシステムに保存されます。次に示すように、tmpfsにキャッシュの場所を作成して、これらをインメモリドライブに変更することをお勧めします。

1. /homeの下に、異なるプロキシキャッシュパス用のディレクトリを作成します。

    たとえば、これらのディレクトリはプライマリFinesse用に作成する必要があります。セカンダリFinesseサーバとCUICサーバでも同じ手順を実行する必要があります。

```
mkdir -p /home/primaryFinesse/rest
mkdir -p /home/primaryFinesse/desktop
mkdir -p /home/primaryFinesse/shindig
```

```
mkdir -p /home/primaryFinesse/openfire
mkdir -p /home/primaryCUIC/cuic
mkdir -p /home/primaryCUIC/cuicdoc
mkdir -p /home/client_temp
mkdir -p /home/proxy_temp
```

```
echo "tmpfs /home/primaryFinesse/rest tmpfs size=1510M,rw,auto,noexec,nodev,nosuid,gid=root,uid=root,mode=1700 0 0" >>
/etc/fstab echo "tmpfs /home/primaryFinesse/desktop tmpfs size=20M,rw,auto,noexec,nodev,nosuid,gid=root,uid=root,mode=1700 0 0"
>>
/etc/fstab echo "tmpfs /home/primaryFinesse/shindig tmpfs size=500M,rw,auto,noexec,nodev,nosuid,gid=root,uid=root,mode=1700 0 0"
>>
/etc/fstab echo "tmpfs /home/primaryFinesse/openfire tmpfs size=10M,rw,auto,noexec,nodev,nosuid,gid=root,uid=root,mode=1700 0 0"
>>
/etc/fstab echo "tmpfs /home/primaryCUIC/cuic tmpfs size=100M,rw,auto,noexec,nodev,nosuid,gid=root,uid=root,mode=1700 0 0" >>
/etc/fstab echo "tmpfs /home/primaryCUIC/cuicdoc tmpfs size=100M,rw,auto,noexec,nodev,nosuid,gid=root,uid=root,mode=1700 0 0"
>>
/etc/fstab echo "tmpfs /home/client_temp tmpfs size=2048M,rw,auto,noexec,nodev,nosuid,gid=root,uid=root,mode=1700 0 0" >>
/etc/fstab echo "tmpfs /home/proxy_temp tmpfs size=2048M,rw,auto,noexec,nodev,nosuid,gid=root,uid=root,mode=1700 0 0" >>
/etc/fstab
```

注：設定に新しいFinesseクラスタを追加するたびに、clientおよびproxy_tempのキャッシュを1 GBずつ増やします。

2. **mount -av**コマンドを使用して、新しいマウントポイントをマウントします。
3. **df -h**コマンドを使用して、ファイルシステムに新しいマウントポイントがマウントされていることを確認します。
4. FinesseおよびCUICキャッシュ設定ファイルのproxy_cache_pathの場所を変更します。

   たとえば、Finesseプライマリのパスを変更するには、<nginx-install-directory>conf/conf.d/finesse/cachesに移動し、既存のキャッシュの場所 /usr/local/openresty/nginx/cache/finesse25/を新しく作成したファイルシステムの場所に変更します /home/primaryFinesse

   ```
   ##Must-change /usr/local/openresty/nginx/cache/finesse25 location would change depending on folder extraction proxy_cache_path
   /home/primaryFinesse/desktop levels=1:2 use_temp_path=on keys_zone=desktop_cache_fin25:10m max_size=15m inactive=3y
   use_temp_path=off; proxy_cache_path /home/primaryFinesse/shindig levels=1:2 use_temp_path=on
   keys_zone=shindig_cache_fin25:10m max_size=500m inactive=3y use_temp_path=off; proxy_cache_path
   /home/primaryFinesse/openfire levels=1:2 use_temp_path=on keys_zone=openfire_cache_fin25:10m max_size=10m inactive=3y
   use_temp_path=off; proxy_cache_path /home/primaryFinesse/rest levels=1:2 use_temp_path=on keys_zone=rest_cache_fin25:10m
   max_size=1500m inactive=40m use_temp_path=off;
   ```

5. Finesseセカンダリサーバ とCUICサーバで同じ手順を実行します。

注：これらのドライブはアプリケーションからはディスクのように見え、多くのメモリ領域

を消費するように構成されたメモリブロックであるため、前のすべての手順で作成したすべてのtmpfsドライブサイズの合計が、導入の最終的なメモリサイジングに追加されていることを確認します。

SSL証明書の設定

自己署名証明書の使用 – テスト展開

自己署名証明書は、逆プロキシを実稼働に展開する準備ができるまで使用する必要があります。実稼働環境では、認証局(CA)署名付き証明書のみを使用します。

1. SSLフォルダコンテンツのNginx証明書を生成します。証明書を生成する前に、/usr/local/openresty/nginxにsslというフォルダを作成する必要があります。これらのコマンドを使用して、2つの証明書を生成する必要があります(1つは<reverseproxy_primary_fqdn>用、もう1つは<reverseproxy_secondary_fqdn>用)。
   a. **sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /usr/local/openresty/nginx/ssl/nginx.key -out /usr/local/openresty/nginx/ssl/nginx.crt**（ホスト名は<reverseproxy_primary_fqdn>として渡します）
   b. **sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /usr/local/openresty/nginx/ssl/nginxnode2.key -out /usr/local/openresty/nginx/ssl/nginxnode2.crt**（ホスト名を：<reverseproxy_secondary_fqdn>として渡します）
   c. 証明書パスが/usr/local/openresty/nginx/ssl/nginx.crtおよび/usr/local/openresty/nginx/ssl/nginxnode2.crtであることを確認します。これらのパスは、Finesse Nginxコンフィギュレーションファイルですでに設定されています。
2. 秘密キー400 (r--------)の権限を変更します。
3. ファイアウォールからの通信を有効にして、Nginxサーバがリッスンするように設定されているポートに対応するように、逆プロキシでファイアウォールとiptablesを設定します。
4. リバースプロキシサーバの/etc/hostsエントリに、Finesse、IdS、およびCUICのIPアドレスとホスト名を追加します。
5. Nginxホストをリバースプロキシとして設定するためにコンポーネントサーバで実行する設定については、ソリューション機能ガイドを参照してください。

注：ここに示す設定は2000年導入の例であり、より大規模な導入に合わせて適切に拡張する必要があります。

CA署名付き証明書の使用 – 実稼働環境

CA署名付き証明書は、次の手順で逆プロキシにインストールできます。

1. 証明書署名要求(CSR)を生成します。

   CSRと秘密キーを生成するには、プロキシにログインした後に入**openssl req -new -newkey rsa:4096 -keyout nginx.key -out nginx.csr**力します。プロンプトに従って、詳細を入力します。これにより、強度4096ビットのCSR（例ではnginx.csr）とRSA秘密キー（例ではnginx.key）が生成されます。

例：

[root@reverseproxyhost.companyname.com ssl]# openssl req -new -newkey rsa:4096 -keyout nginx.key -out nginx.csr Generating a RSA private key .....+++++ .....+++++ writing new private key to 'nginx.key' Enter PEM pass phrase:passphrase Verifying - Enter PEM pass phrase:passphrase ----- You are about to be asked to enter information that will be incorporated into your certificate request. What you are about to enter is what is called a Distinguished Name or a DN. There are quite a few fields but you can leave some blank For some fields there will be a default value, If you enter '.', the field will be left blank. ----- Country Name (2 letter code) [XX]:US State or Province Name (full name) []:CA Locality Name (eg, city) [Default City]:Orange County Organization Name (eg, company) [Default Company Ltd]:CompanyName Organizational Unit Name (eg, section) []:BusinessUnit Common Name (eg, your name or your server's hostname) []:reverseproxyhostname.companydomain.com Email Address []:john.doe@comapnydomain.com Please enter the following 'extra' attributes to be sent with your certificate request A challenge password []:challengePWD An optional company name []:CompanyName

PEMパスフレーズを書き留めておきます。これは、導入時に秘密キーを復号化するために使用されます。

2. CAから署名付き証明書を取得します。

   CSRを認証局(CA)に送信し、署名付き証明書を取得します。

   注：CAから受信した証明書が、対応するすべての証明書を含む証明書チェーンではない場合、関連するすべての証明書を1つの証明書チェーンファイルに構成します。

3. 証明書とキーを展開します。

   最初の手順の一部として前に生成したキーをコマンドで復号化**openssl rsa -in nginx.key -out nginx_decrypted.key**します。逆プロキシマシンのフォルダ/usr/local/openresty/nginx/ssl内にCA署名付き証明書と復号化されたキーを配置します。コンフィギュレーションファイル /usr/local/openresty/nginx/conf/conf.d/ssl/ssl.confのNginx設定の証明書に関連するSSL設定を更新または追加します。

ssl_certificate /usr/local/openresty/nginx/ssl/ca_signed_cert.crt; ssl_certificate_key /usr/local/openresty/nginx/ssl/nginx_decrypted.key;

4. 証明書のアクセス許可を設定します。

   chmod 400 /usr/local/openresty/nginx/ssl/ca_signed_cert.crtと入力し**chmod 400 /usr/local/openresty/nginx/ssl/nginx_decrypted.key**、証明書に読み取り専用権限を与え、所有者に制限します。

5. Nginxをリロードします。

カスタムDiffie-Hellmanパラメータの使用

次のコマンドを使用して、カスタムDiffie-Hellmanパラメータを作成します。

openssl dhparam -out /usr/local/openresty/nginx/ssl/dhparam.pem 2048 chmod 400 /usr/local/openresty/nginx/ssl/dhparam.pem

/usr/local/openresty/nginx/conf/conf.d/ssl/ssl.confファイルの新しいパラメータを使用するように、サーバ設定を変更します。

ssl_dhparam /usr/local/openresty/nginx/ssl/dhparam.pem;

OCSPホチキス止めが有効になっていることの確認：証明書失効チェック

注：これを有効にするには、サーバはCA署名付き証明書を使用する必要があり、サーバは証明書に署名したCAにアクセスできる必要があります。

file/usr/local/openresty/nginx/conf/conf.d/ssl/ssl.confで次の設定を追加または更新します。

ssl_stapling on; ssl_stapling_verify on;

Nginxの設定

セキュリティを強化し、パフォーマンスを向上させるために、デフォルトのNginx設定ファイル(/usr/local/openresty/nginx/conf/nginx.conf)をこれらのエントリを含むように変更する必要があります。この内容は、Nginxのインストールによって作成されるデフォルトのコンフィギュレーションファイルを変更するために使用する必要があります。

```
# Increasing number of worker processes will not increase the processing the request. The number of wor
# in system CPU. Nginx provides "auto" option to automate this, which will spawn one worker for each CP
worker_processes  auto;

# Process id file location
pid /usr/local/openresty/nginx/logs/nginx.pid;

# Binds each worker process to a separate CPU
worker_cpu_affinity auto;

#Defines the scheduling priority for worker processes. This should be calculated by "nice" command. In
worker_priority 0;


error_log  /usr/local/openresty/nginx/logs/error.log info;

#user root root;

# current limit on the maximum number of open files by worker processes, keeping 10 times of worker_con

worker_rlimit_nofile 102400;

events {
    multi_accept on;


    # Sets the maximum number of simultaneous connections that can be opened by a worker process.
```

```
    # This should not be more the current limit on the maximum number of open files i.e. hard limit of
    # The appropriate setting depends on the size of the server and the nature of the traffic, and can
    worker_connections  10240;
    #debug_connection 10.78.95.21
}


http {

    include       mime.types;


    default_type  text/plain;


    ## Must-change Change with DNS resolver ip in deployment
    resolver 192.168.1.3;


    ## Must-change change lua package path to load lua libraries
    lua_package_path  "/usr/local/openresty/lualib/resty/?.lua;/usr/local/openresty/nginx/lua/?.lua;;"


    ## Must-change change proxy_temp folder as per cache directory configurations
    proxy_temp_path /usr/local/openresty/nginx/cache/proxy_temp 1 2 ;
    ## Must-change change client_temp folder as per cache directory configurations
    client_body_temp_path /usr/local/openresty/nginx/cache/client_temp 1 2 ;


    lua_shared_dict userlist 50m;
    lua_shared_dict credentialsstore 100m;
    lua_shared_dict userscount 100k;
    lua_shared_dict clientstorage 100m;
    lua_shared_dict blockingresources 100m;
    lua_shared_dict tokencache_saproxy 10M;
    lua_shared_dict tokencache_saproxy125 10M;
    lua_shared_dict ipstore 10m;
    lua_shared_dict desktopurllist 10m;
    lua_shared_dict desktopurlcount 100k;
    lua_shared_dict thirdpartygadgeturllist 10m;
    lua_shared_dict thirdpartygadgeturlcount 100k;
    lua_shared_dict corsheadersstore 100k;


    init_worker_by_lua_block {
        local UsersListManager = require('users_list_manager')
        local UnauthenticatedDesktopResourcesManager = require("unauthenticated_desktopresources_manage
        local UnauthenticatedResourcesManager = require("unauthenticated_thirdpartyresources_manager")
        --  Must-change Replace saproxy.cisco.com with reverseproxy fqdn

        if  ngx.worker.id() == 0 then
            UsersListManager.getUserList("saproxy.cisco.com", "https://saproxy.cisco.com:8445/finesse/a
            UnauthenticatedDesktopResourcesManager.getDesktopResources("saproxy.cisco.com", "https://sa
            UnauthenticatedResourcesManager.getThirdPartyGadgetResources("saproxy.cisco.com", "https://
        end
    }

    include conf.d/*.conf;

    sendfile        on;

    tcp_nopush     on;
```

```
server_names_hash_bucket_size 512;
```

リバースプロキシポートの設定

デフォルトでは、Nginx設定はポート8445でFinesse要求をリッスンします。同時に、Finesse要求をサポートするためにリバースプロキシから有効にできるポートは1つだけです（8445など）。ポート443をサポートする必要がある場合は、<nginx-install-directory>conf/conf.d/finesse.confファイルを編集して、443でのリスニングを有効にし、8445でのリスニングを無効にします。

逆プロキシコンポーネントとアップストリームコンポーネント間の相互TLS認証の設定

逆プロキシホストからの接続用のクライアントSSL証明書認証は、新しいCVOS CLIオプションを使用して、CCBUアップストリームコンポーネントのCUIC/Finesse/IdS/Livedataで有効にできます。

utils system reverse-proxy client-auth enable/disable/statusコマンドを発行します。

デフォルトではこれは無効になっており、各アップストリームサーバで個別にCLIを実行することにより、管理者が明示的に有効にする必要があります。このオプションを有効にすると、アップストリームホストで実行されているCisco Web Proxy Service(WPS)は、CLI utils system reverse-proxy allowed-hosts add <proxy-host>の一部として追加された信頼できる逆プロキシホストから発信された接続のTLSハンドシェイクでのクライアント証明書の認証を開始します。

プロキシ設定ファイル(ssl.confおよびssl2.conf)での同じ設定ブロックを次に示します。

#Must-change /usr/local/openresty/nginx/ssl/nginx.crt change this location accordingly proxy_ssl_certificate /usr/local/openresty/nginx/ssl/nginx.crt; #Must-change /usr/local/openresty/nginx/ssl/nginx.key change this location accordingly proxy_ssl_certificate_key /usr/local/openresty/nginx/ssl/nginx.key;

発信トラフィック（プロキシからアップストリーム）に使用するSSL証明書は、着信トラフィック（コンポーネントサーバブロック用のSSLコネクタ）に設定したSSL証明書と同じにすることができます。自己署名証明書がproxy_ssl_certificateとして使用される場合、証明書が正常に認証されるために、アップストリームコンポーネント(Finesse/IdS/CUIC/Livedata)のtomcat信頼ストアにアップロードされる必要があります。

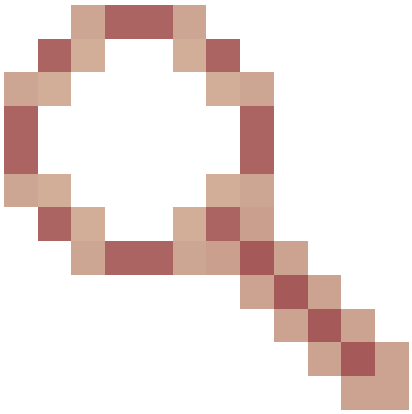
逆プロキシによるアップストリームサーバ証明書の検証はオプションであり、デフォルトでは無効になっています。逆プロキシホストとアップストリームホストの間で完全なTLS相互認証を実現する場合は、ssl.confおよびssl2.confファイルから次の設定をコメントアウトする必要があります。

#Enforce upstream server certificate validation at proxy -> #this is not mandated as per CIS buit definitely adds to security. #It requires the administrator to upload all upstream server certificates to the proxy certificate store #Must-Change Uncomment below lines IF need to enforce upstream server certificate validation at proxy #proxy_ssl_verify on; #proxy_ssl_trusted_certificate /usr/local/openresty/nginx/ssl/finesse25.crt;

proxy_ssl_trusted_certificate: This file should contain the all upstream certificate enteries concatenated together

相互TLS認証の設定に関する注意事項：

- この機能をCCBUコンポーネントで有効にすると、TLSハンドシェイク時にもLANクライアントからクライアント証明書が要求されます。クライアントマシンにクライアント/個人証明書がインストールされている場合、ブラウザは、クライアント認証に適切な証明書を選択するよう求めるポップアップをエンドユーザに表示できます。どの証明書エンドユーザが選択するか、またはポップアップ要求でキャンセルを押すかは問題ではありませんが、クライアント証明書認証はLANクライアントには適用されず、エクスペリエンスの変更が行われる



ため、成功します。CDET CSCwa26057を参照してください。
を参照してください。
- webproxyサービスで解決できないallowed-listにプロキシホストが追加された場合、アップストリームコンポーネントのwebproxyサービスが起動しません。許可リストに追加された逆プロキシホストが、DNSルックアップによってアップストリームコンポーネントから解決可能であることを確認します。

キャッシュのクリア

逆プロキシキャッシュをクリアするには、

/clearCache.sh
コマンドを使用します。

標準ガイドライン

ここでは、Nginxをプロキシサーバとして設定するときに従う必要がある標準ガイドラインについて簡単に説明します。

これらのガイドラインは、Center for Internet Securityから派生したものです。各ガイドラインの詳細については、同じドキュメントを参照してください。

1. 常に最新の安定したOpenRestyおよびOpenSSLバージョンを使用することをお勧めします。
2. Nginxは別のディスクマウントにインストールすることをお勧めします。
3. Nginxプロセスは、ルートユーザが所有し（または選択したOSに適用可能）、権限644(rw-------)またはそれ以上の権限が必要です。

4. Nginxは不明なホストに対する要求をブロックする必要があります。各サーバー・ブロックに、明示的に定義されたserver_nameディレクティブが含まれていることを確認します。確認するには、nginx.confおよびnginx/conf.dディレクトリ内のすべてのサーバブロックを検索し、すべてのサーバブロックにserver_nameが含まれていることを確認します。
5. Nginxは許可されたポートだけをリッスンする必要があります。nginx.confおよびnginx/conf.dディレクトリ内のすべてのサーバブロックを検索し、listen toディレクティブを確認して、認可ポートだけがlistening用に開かれていることを確認します。
6. Cisco FinesseはHTTPをサポートしていないため、プロキシサーバのHTTPポートもブロックすることをお勧めします。
7. Nginx SSLプロトコルはTLS 1.2である必要があります。従来のSSLプロトコルのサポートは削除する必要があります。また、脆弱なSSL暗号を無効にする必要もあります。
8. Nginxエラーとアクセスログをリモートsyslogサーバに送信することを推奨します。
9. Webアプリケーションファイアウォールとして動作するmod_securityモジュールをインストールすることを推奨します。詳細は、『ModSecurityマニュアル』を参照してください。Nginxのロードは、所定のmod_securityモジュール内では検証されないことに注意してください。

マッピングファイルの設定

Finesseデスクトップのリバースプロキシ導入では、外部から見えるホスト名とポートの組み合わせのリストを設定し、それらの組み合わせをFinesse、IdS、およびCUICサーバで使用される実際のサーバ名とポートにマッピングするためのマッピングファイルが必要です。内部サーバで設定されるこのマッピングファイルは、インターネット経由で接続されたクライアントを、インターネットで使用される必要なホストとポートにリダイレクトできるようにする主要な設定です。

マッピングファイルは、コンポーネントサーバにアクセス可能なWebサーバに展開する必要があり、展開が機能するようにそのURIを設定する必要があります。マッピングファイルは、ネットワーク内で使用可能な専用Webサーバを使用して設定することをお勧めします。このようなサーバが使用できない場合は、代わりに逆プロキシを使用できます。逆プロキシを使用するには、ネットワーク内からプロキシにアクセスできる必要があり、DMZに不正にアクセスできる外部クライアントに情報が公開されるリスクがあります。次のセクションでは、これを実現する方法について詳しく説明します。

すべてのコンポーネントサーバでマッピングファイルURIを設定する正確な手順、およびマッピングファイルデータの作成方法の詳細については、機能ガイドを参照してください。

マッピングファイルサーバとして逆プロキシを使用

これらの手順は、逆プロキシがプロキシマッピングファイルホストとしても使用されている場合にのみ必要です。

1. Finesse/CUICおよびIdSホストが使用するドメインコントローラで、そのIPアドレスが解決されるように逆プロキシホスト名を設定します。
2. 生成されたNginx署名付き証明書をcmplatformのtomcat-trustの下の両方のノードにアップロードし、サーバを再起動します。
3. <NGINX_HOME>/html/proxymap.txtでMust-changeの値を更新します。
4. **nginx -s reload**コマンドを使用してNginx設定をリロードします。

5. curlコマンドを使用して、別のネットワークホストからコンフィギュレーションファイルにアクセスできることを確認する。

CentOS 8カーネルの強化

選択したオペレーティングシステムがCentOS 8である場合、プロキシをホストする専用サーバを使用するインストールでは、これらのsysctl設定を使用してカーネルの強化/調整を行うことをお勧めします。

```
## Configurations for kernel hardening - CentOS8. The file path is /etc/sysctl.conf
## Note that the commented configurations denote that CentOS 8's default value matches
## the recommended/tested value, and are not security related configurations.

# Avoid a smurf attack
net.ipv4.icmp_echo_ignore_broadcasts = 1

# Turn on protection for bad icmp error messages
net.ipv4.icmp_ignore_bogus_error_responses = 1

# Turn on syncookies for SYN flood attack protection
net.ipv4.tcp_syncookies = 1

# Turn on and log spoofed, source routed, and redirect packets
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.default.log_martians = 1

# Turn off routing
net.ipv4.ip_forward = 0
net.ipv4.conf.all.forwarding = 0
net.ipv6.conf.all.forwarding = 0

net.ipv4.conf.all.mc_forwarding = 0
net.ipv6.conf.all.mc_forwarding = 0

# Block routed packets
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.default.accept_source_route = 0
net.ipv6.conf.all.accept_source_route = 0
net.ipv6.conf.default.accept_source_route = 0

# Block ICMP redirects
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.default.accept_redirects = 0
net.ipv4.conf.all.secure_redirects = 0
net.ipv4.conf.default.secure_redirects = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.send_redirects = 0

# Filter routing packets with inward-outward path mismatch(reverse path filtering)
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.rp_filter = 1

# Router solicitations & advertisements related.
net.ipv6.conf.default.router_solicitations = 0
net.ipv6.conf.default.accept_ra_rtr_pref = 0
```

```
net.ipv6.conf.default.accept_ra_pinfo = 0
net.ipv6.conf.default.accept_ra_defrtr = 0
net.ipv6.conf.default.autoconf = 0
net.ipv6.conf.default.dad_transmits = 0
net.ipv6.conf.default.max_addresses = 1
net.ipv6.conf.all.accept_ra = 0
net.ipv6.conf.default.accept_ra = 0

# Backlog - increased from default 1000 to 5000.
net.core.netdev_max_backlog = 5000

# Setting syn/syn-ack retries to zero, so that they don't stay in the queue.
net.ipv4.tcp_syn_retries = 0
net.ipv4.tcp_synack_retries = 0

# Max tcp listen backlog. Setting it to 511 to match nginx config
net.core.somaxconn = 511

# Reduce the duration of connections held in TIME_WAIT(seconds)
net.ipv4.tcp_fin_timeout = 6

# Maximum resources allotted
# fs.file-max = 2019273
# kernel.pid_max = 4194304
# net.ipv4.ip_local_port_range = 32768 60999

# TCP window size tuning
# net.ipv4.tcp_window_scaling = 1
# net.core.rmem_default = 212992
# net.core.rmem_max = 212992
# net.ipv4.tcp_rmem = 4096 87380 6291456
# net.ipv4.udp_rmem_min = 4096
# net.core.wmem_default = 212992
# net.core.wmem_max = 212992
# net.ipv4.tcp_wmem = 4096 16384 4194304
# net.ipv4.udp_wmem_min = 4096
# vm.lowmem_reserve_ratio = 256 256 32 0 0
# net.ipv4.tcp_mem = 236373 315167 472746

# Randomize virtual address space
kernel.randomize_va_space = 2

# Congestion control
# net.core.default_qdisc = fq_codel
# net.ipv4.tcp_congestion_control = cubic

# Disable SysReq
kernel.sysrq = 0

# Controls the maximum size of a message, in bytes
kernel.msgmnb = 65536

# Controls the default maximum size of a message queue
kernel.msgmax = 65536

# Controls the eagerness of the kernel to swap.
vm.swappiness = 1
```

推奨される変更を行った後は、リブートすることをお勧めします。

IPテーブルの強化

IPtablesは、システム管理者がLinuxカーネルファイアウォールによって提供されるIPv4とIPv6のテーブル、チェーン、およびルールを設定できるようにするアプリケーションです。

これらのIPtableルールは、Linuxカーネルファイアウォールのアクセスを制限することにより、プロキシアプリケーションを総当たり攻撃から保護するように設定されています。

設定内のコメントは、ルールを使用してレート制限されているサービスを示します。

注：管理者が異なるポートを使用するか、同じポートを使用する複数のサーバへのアクセスを拡大する場合は、これらのポートに対し、これらの数に基づいて適切なサイジングを行う必要があります。

```
## Configuration for iptables service
## The file path is /etc/sysconfig/iptables
## Make a note for must-change values to be replaced.
## Restart of the iptable service is required after applying following rules

*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]

# Ensure loopback traffic is configured
-A INPUT -i lo -j ACCEPT
-A OUTPUT -o lo -j ACCEPT
-A INPUT -s 127.0.0.0/8 -j DROP

# Ensure ping openeded only for the particular source and blocked for rest
# Must-Change: Replace the x.x.x.x with valid ip address
-A INPUT -p ICMP --icmp-type 8 -s x.x.x.x -j ACCEPT

# Ensure outbound and established connections are configured
-A INPUT -p tcp -m state --state RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -p tcp -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT

# Block ssh for external interface
# Must-Change: Replace the ens224 with valid ethernet interface
-A INPUT -p tcp -i ens224 --dport 22 -j DROP
# Open inbound ssh(tcp port 22) connections
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT


# Configuration for finesse 8445 port
-A INPUT -p tcp -m tcp --dport 8445 --tcp-flags SYN SYN -m connlimit --connlimit-above 10 --connlimit-ma
-A INPUT -p tcp -m tcp --dport 8445 --tcp-flags SYN SYN -m connlimit --connlimit-above 10 --connlimit-ma
 -A INPUT -p tcp -m tcp --dport 8445 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 6/sec --hashlimi
-A INPUT -p tcp -m tcp --dport 8445 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG -
-A INPUT -p tcp -m tcp --dport 8445 --tcp-flags SYN SYN -j DROP

# Configuration for IdS 8553 port
-A INPUT -p tcp -m tcp --dport 8553 --tcp-flags SYN SYN -m connlimit --connlimit-above 6 --connlimit-ma
-A INPUT -p tcp -m tcp --dport 8553 --tcp-flags SYN SYN -m connlimit --connlimit-above 6 --connlimit-ma
-A INPUT -p tcp -m tcp --dport 8553 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 2/sec --hashlimit
```

```
-A INPUT -p tcp -m tcp --dport 8553 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG -
-A INPUT -p tcp -m tcp --dport 8553 --tcp-flags SYN SYN -j DROP

# Configuration for IdP 443 port
-A INPUT -p tcp -m tcp --dport 443 --tcp-flags SYN SYN -m connlimit --connlimit-above 8 --connlimit-mas
-A INPUT -p tcp -m tcp --dport 443 --tcp-flags SYN SYN -m connlimit --connlimit-above 8 --connlimit-mas
-A INPUT -p tcp -m tcp --dport 443 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 4/sec --hashlimit-
-A INPUT -p tcp -m tcp --dport 443 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG --
-A INPUT -p tcp -m tcp --dport 443 --tcp-flags SYN SYN -j DROP


# Must-Change: A2A file transfer has not been considered for below IMNP configuration.
# For A2A for support, these configuration must be recalculated to cater different file transfer scenar

# Configuration for IMNP 5280 port
-A INPUT -p tcp -m tcp --dport 5280 --tcp-flags SYN SYN -m connlimit --connlimit-above 30 --connlimit-m
-A INPUT -p tcp -m tcp --dport 5280 --tcp-flags SYN SYN -m connlimit --connlimit-above 30 --connlimit-m
-A INPUT -p tcp -m tcp --dport 5280 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 20/sec --hashlimi
-A INPUT -p tcp -m tcp --dport 5280 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG -
-A INPUT -p tcp -m tcp --dport 5280 --tcp-flags SYN SYN -j DROP

# Configuration for IMNP 15280 port
-A INPUT -p tcp -m tcp --dport 15280 --tcp-flags SYN SYN -m connlimit --connlimit-above 30 --connlimit-
-A INPUT -p tcp -m tcp --dport 15280 --tcp-flags SYN SYN -m connlimit --connlimit-above 30 --connlimit-
-A INPUT -p tcp -m tcp --dport 15280 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 20/sec --hashlim
-A INPUT -p tcp -m tcp --dport 15280 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG
-A INPUT -p tcp -m tcp --dport 15280 --tcp-flags SYN SYN -j DROP


# Configuration for IMNP 25280 port
-A INPUT -p tcp -m tcp --dport 25280 --tcp-flags SYN SYN -m connlimit --connlimit-above 30 --connlimit-
-A INPUT -p tcp -m tcp --dport 25280 --tcp-flags SYN SYN -m connlimit --connlimit-above 30 --connlimit-
-A INPUT -p tcp -m tcp --dport 25280 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 20/sec --hashlim
-A INPUT -p tcp -m tcp --dport 25280 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG
-A INPUT -p tcp -m tcp --dport 25280 --tcp-flags SYN SYN -j DROP


# Configuration for CUIC 8444 port
-A INPUT -p tcp -m tcp --dport 8444 --tcp-flags SYN SYN -m connlimit --connlimit-above 6 --connlimit-ma
-A INPUT -p tcp -m tcp --dport 8444 --tcp-flags SYN SYN -m connlimit --connlimit-above 6 --connlimit-ma
-A INPUT -p tcp -m tcp --dport 8444 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 2/sec --hashlimit
-A INPUT -p tcp -m tcp --dport 8444 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG -
-A INPUT -p tcp -m tcp --dport 8444 --tcp-flags SYN SYN -j DROP

# Configuration for CUIC 8447 port
-A INPUT -p tcp -m tcp --dport 8447 --tcp-flags SYN SYN -m connlimit --connlimit-above 6 --connlimit-ma
-A INPUT -p tcp -m tcp --dport 8447 --tcp-flags SYN SYN -m connlimit --connlimit-above 6 --connlimit-ma
-A INPUT -p tcp -m tcp --dport 8447 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 2/sec --hashlimit
-A INPUT -p tcp -m tcp --dport 8447 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG -
-A INPUT -p tcp -m tcp --dport 8447 --tcp-flags SYN SYN -j DROP

# Configuration for LiveData 12005  port
-A INPUT -p tcp -m tcp --dport 12005 --tcp-flags SYN SYN -m connlimit --connlimit-above 10 --connlimit-
-A INPUT -p tcp -m tcp --dport 12005 --tcp-flags SYN SYN -m connlimit --connlimit-above 10 --connlimit-
-A INPUT -p tcp -m tcp --dport 12005 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 6/sec --hashlimi
-A INPUT -p tcp -m tcp --dport 12005 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG
-A INPUT -p tcp -m tcp --dport 12005 --tcp-flags SYN SYN -j DROP

# Configuration for LiveData 12008  port
-A INPUT -p tcp -m tcp --dport 12008 --tcp-flags SYN SYN -m connlimit --connlimit-above 10 --connlimit-
-A INPUT -p tcp -m tcp --dport 12008 --tcp-flags SYN SYN -m connlimit --connlimit-above 10 --connlimit-
-A INPUT -p tcp -m tcp --dport 12008 --tcp-flags SYN SYN -m hashlimit --hashlimit-upto 6/sec --hashlimi
```

```
-A INPUT -p tcp -m tcp --dport 12008 --tcp-flags SYN SYN -m limit --limit 1/min --limit-burst 1 -j LOG
-A INPUT -p tcp -m tcp --dport 12008 --tcp-flags SYN SYN -j DROP

# Block all other ports
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited

COMMIT
```

これらの規則は、/etc/sysconfig/iptablesを手動で編集して直接適用するか、または設定をiptables.confなどのファイルに保存し、 cat iptables.conf >>/etc/sysconfig/iptablesを実行して適用できます。

ルールを適用した後、IPtablesサービスを再起動する必要があります。IPtablesサービスを再起動するには、**systemctl restart iptables**を入力します。

クライアント接続の制限

以前のIPtable設定に加えて、プロキシを使用するクライアントのアドレス範囲を知っているインストールでは、この知識を使用してプロキシアクセスルールを保護することを推奨します。これは、オンラインセキュリティに関して緩やかなルールを持つ国のIPアドレスの範囲で作成されることが多い悪意のあるネットワークのボットネットからプロキシを保護することに関しては、大きな見返りがあります。したがって、アクセスパターンが確実である場合は、IPアドレス範囲を国/州またはISPベースのIP範囲に制限することを強く推奨します。

クライアント接続のブロック

また、攻撃がIPアドレスまたはIPアドレスの範囲から行われることが特定された場合に、特定の範囲のアドレスをブロックする方法を知ることも役立ちます。このような場合、これらのIPアドレスからの要求はiptableルールでブロックできます。

個別のIPアドレスのブロック

複数の異なるIPアドレスをブロックするには、各IPアドレスのIPTables設定ファイルに1行を追加します。

たとえば、アドレス192.0.2.3と192.0.2.4をブロックするには、次のように入力します。

<#root>

```
iptables -A INPUT -s
```

192.0.2.3

```
 -j DROP iptables -A INPUT -s
```

192.0.2.4

```
 - j DROP.
```

IPアドレスの範囲のブロック

範囲内の複数のIPアドレスをブロックし、IP範囲を指定してIPTables設定ファイルに1行を追加します。

たとえば、192.0.2.3 ~ 192.0.2.35のアドレスをブロックするには、次のように入力します。

```
iptables -A INPUT -m iprange --src-range 192.0.2.3-192.0.2.35 -j DROP.
```

サブネット内のすべてのIPアドレスをブロックする

サブネット全体のすべてのIPアドレスをブロックするには、クラスレスドメイン間ルーティング(CIDR)表記をIPアドレス範囲に使用して、IPTablesコンフィギュレーションファイルに1行を追加します。たとえば、すべてのクラスCアドレスをブロックするには、次のように入力します。

```
iptables -A INPUT -s 192.0.0.0/16 -j DROP.
```

SELinux

SELinuxは、Linux OSの拡張機能として統合されたプラットフォームセキュリティフレームワークです。逆プロキシとしてOpenRestyを実行するためにSELinuxポリシーをインストールして追加する手順は、次のとおりです。

1. **openresty -s stop**コマンドを使用してプロセスを停止します。
2. **systemctl**コマンドを使用して/stop nginx serverを設定および開始します。これにより、起動中にOpenRestyプロセスが自動的に開始されます。rootユーザとして次のコマンドを入力します。
    a. /usr/lib/systemd/system に移動します。
    b. openresty.serviceというファイルを開きます。
    c. PIDFileの場所に従ってファイルの内容を更新します。

```
[Unit]
Description=The OpenResty Application Platform
After=syslog.target network-online.target remote-fs.target nss-lookup.target
Wants=network-online.target

[Service]
Type=forking
PIDFile=/usr/local/openresty/nginx/logs/nginx.pid
ExecStartPre=/usr/local/openresty/nginx/sbin/nginx -t
ExecStart=/usr/local/openresty/nginx/sbin/nginx
ExecReload=/bin/kill -s HUP $MAINPID
ExecStop=/bin/kill -s QUIT $MAINPID
PrivateTmp=true
```

```
[Install]
WantedBy=multi-user.target
```

d. rootユーザで、**sudo systemctl enable openresty**と入力します。

e. **systemctl start openresty / systemctl stop openresty**コマンドを使用してOpenRestyサービスを開始/停止し、プロセスがrootユーザとして開始/停止することを確認します。

1. Selinuxのインストール
   - デフォルトでは、一部のSELinuxパッケージだけがCentOsにインストールされます。
   - SELinuxポリシーを生成するには、policycoreutils-develパッケージとその依存関係がインストールされている必要があります。
   - policycoreutils-develをインストールするには、次のコマンドを入力します

```
yum install policycoreutils-devel
```

   - パッケージのインストール後に「**sepolicy**」コマンドが機能することを確認します。

```
usage: sepolicy [-h] [-P POLICY]

                {booleans,communicate,generate,gui,interface,manpage,network,transition}
                ...

SELinux Policy Inspection Tool
```

2. 新しいLinuxユーザの作成とSElinuxユーザとのマッピング

a. LinuxユーザとSELinuxユーザのマッピングを表示するには、 **semanage login -l**を入力します。

```
[root@loadproxy-cisco-com ~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *              *
root                 unconfined_u         s0-s0:c0.c1023       *
```

b. rootとして、SELinux user_uユーザにマッピングされる新しいLinuxユーザ(nginxユーザ)を作成します。

```
useradd -Z user_u nginxuser
[root@loadproxy-cisco-com ~]# passwd nginxuser
Changing password for user nginxuser.
New password:
```

```
Retype new password:
passwd: all authentication tokens updated successfully.
```

c. nginxuserとuser_uの間のマッピングを表示するには、rootとして次のコマンドを入力します。

```
[root@loadproxy-cisco-com ~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
nginxuser            user_u               s0                   *
root                 unconfined_u         s0-s0:c0.c1023       *
```

d. SELinux __default__ loginは、デフォルトでSELinux unconfinded_uユーザにマッピングされます。次のコマンドを使用して、デフォルトでuser_uを閉じ込める必要があります。

```
semanage login -m -s user_u -r s0 __default__
```

コマンドが正しく機能したかどうかを確認するには、**semanage login -l**と入力します。次の出力が生成されます。

```
Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0                   *
nginxuser            user_u               s0                   *
root                 unconfined_u         s0-s0:c0.c1023       *
```

e. nginx.confを変更し、nginxuserの所有権変更を実行します。

i. <Openresty-install-directory>ディレクトリで「**chown -R nginxuser:nginxuser**」と入力します。

ii. nginx.confファイルを修正し、ワーカープロセスを実行するユーザーとしてnginxuserを含めます。

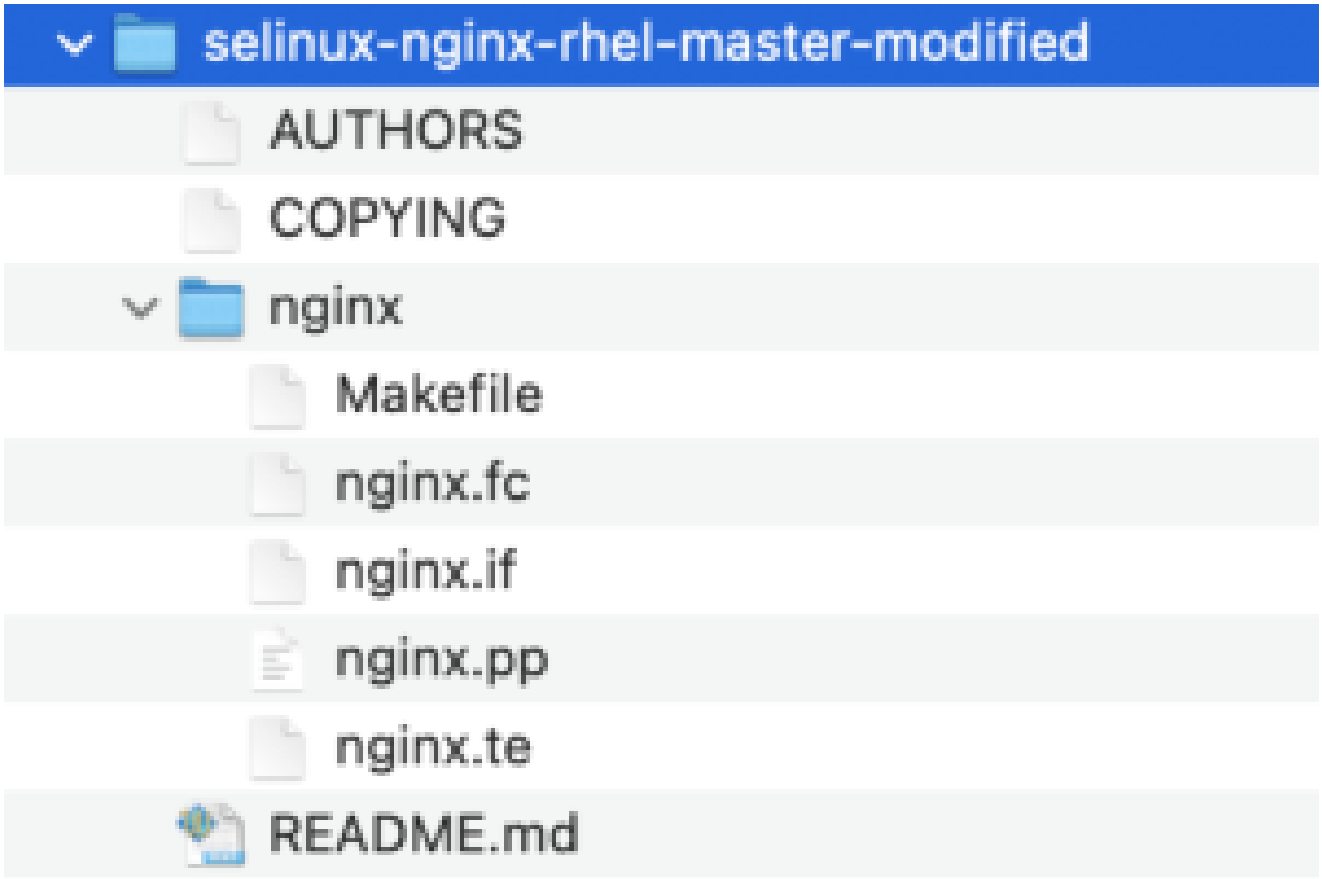
```
........

user nginxuser nginxuser;

..........
```

Nginx用のSELinuxポリシーの作成

1. **sepolicy generate --init /usr/bin/nginx**コマンドを使用してNginx用の新しいデフォルトのカスタムポリシーを生成する代わりに、既存のポリシーで開始することを推奨します。
2. 指定されたURLからダウンロードするnginx.fcファイル（ファイルコンテキストファイル）およびnginx.te（型強制ファイル）ファイルが、リバースプロキシの使用に適合するように変更されました。
3. この修正バージョンは、特定の使用例に対して修正されているため、参照用として使用できます。
4. file software download pageからファイルselinux-nginx-rhel-master-modified.tarをダウンロードします。



5. .tarファイルを抽出し、その中のnginxディレクトリに移動します。
6. .fcファイルを開き、nginxインストーラ、キャッシュ、およびpidファイルの必要なファイルパスを確認します。
7. **make**コマンドを使用して設定をコンパイルします。
8. nginx.ppファイルが生成されます。
9. **semodule**コマンドを使用してポリシーをロードします。

```
semodule -i nginx.pp
```

10. /rootに移動し、touch /.autorelabelという名前の空のファイルを作成します。
11. システムをリブートします。

12. ポリシーが正常にロードされたことを確認するには、次のコマンドを入力します。

```
semodule --list-modules=full
```

```
[root@loadproxy-cisco-com ~]# semodule --list-modules=full
400 nginx                   pp
200 container               pp
200 flatpak                 pp
100 abrt                    pp
100 accountsd               pp
100 acct                    pp
100 afs                     pp
100 aiccu                   pp
100 aide                    pp
100 ajaxterm                pp
100 alsa                    pp
```

13. Nginxは違反なしで実行する必要があります。(違反は/var/log/messagesおよび/var/log/audit/audit.logで確認できます)。
14. Nginxのステータスを確認するには、次のコマンドを入力します。

```
ps -aefZ | grep nginx
```

```
[root@loadproxy-cisco-com ~]# ps -aefZ |grep nginx
system_u:system_r:nginx_t:s0    root        1686       1  0 16:14 ?        00:00:00 nginx: master process /usr/bin/nginx
system_u:system_r:nginx_t:s0    nginxus+    1687    1686  0 16:14 ?        00:00:00 nginx: worker process
system_u:system_r:nginx_t:s0    nginxus+    1688    1686  0 16:14 ?        00:00:00 nginx: worker process
system_u:system_r:nginx_t:s0    nginxus+    1689    1686  0 16:14 ?        00:00:00 nginx: worker process
system_u:system_r:nginx_t:s0    nginxus+    1690    1686  0 16:14 ?        00:00:00 nginx: worker process
system_u:system_r:nginx_t:s0    nginxus+    1691    1686  0 16:14 ?        00:00:00 nginx: worker process
system_u:system_r:nginx_t:s0    nginxus+    1692    1686  0 16:14 ?        00:00:00 nginx: worker process
system_u:system_r:nginx_t:s0    nginxus+    1693    1686  0 16:14 ?        00:00:00 nginx: worker process
system_u:system_r:nginx_t:s0    nginxus+    1694    1686  0 16:14 ?        00:00:00 nginx: worker process
system_u:system_r:nginx_t:s0    nginxus+    1695    1686  0 16:14 ?        00:00:00 nginx: cache manager process
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 root 2543 2252  0 16:17 pts/0 00:00:00 grep --color=auto nginx
```

15. これで、Finesseエージェント/スーパーバイザデスクトップにアクセスできるようになります。

確認

ここでは、設定が正常に機能しているかどうかを確認します。

Finesse

1. DMZからhttps://<reverseproxy:port>/finesse/api/SystemInfo.を要求し、それらが到達可能であるかどうかを確認します。
2. <primaryNode>と<secondaryNode>の両方の<host>の値が有効な逆プロキシホスト名であることを確認します。Finesseホスト名は使用できません。

CUICおよびライブデータ

1. 応答に逆プロキシホスト名ではなくFinesseホスト名が表示される場合、『Finesse 12.6 UCCE Feature Guide』の「VPNを使用しないFinesseデスクトップへのアクセス」の「ネットワーク変換データの入力」の項の説明に従って、プロキシマッピング設定と許可されているホストがFinesseサーバに正しく追加されていることを確認します。
2. LiveDataガジェットがFinesseデスクトップで正しくロードされる場合、CUICとLiveDataプロキシの設定は適切です。
3. CUICとLiveDataの設定を検証するには、DMZからこれらのURLに対してHTTP要求を行い、それらが到達可能であるかどうかを確認します。
   - https://<reverseproxy:cuic_port>/cuic/rest/about
   - https://<reverseproxy:ldweb_port>/livedata/security
   - https://<reverseproxy:ldsocketio_port>/security

IDS

IdS設定を検証するには、次の手順を実行します。

1. 管理インターフェイスは逆プロキシ経由では公開されないため、LANからIdSAdminインターフェイス(https://<ids_LAN_host:ids_port>:8553/idsadmin)にログインします。
2. Settings > IdS Trustの順に選択します。
3. プロキシクラスタパブリッシャノードがDownload SP metadataページにリストされていることを確認し、Nextをクリックします。
4. Upload IDP metadataページでIDPプロキシが正しく表示されるように設定されていることを確認し、Nextをクリックします。
5. テストSSOページからすべてのプロキシクラスタノードを介してテストSSOを開始し、すべてが正常に実行されていることを検証します。これには、逆プロキシノードへのクライアントマシン接続が必要です。

パフォーマンス

nmonツールを使用した、同等の上位パフォーマンスキャプチャのデータ分析は、『Finesse Release 12.6(1) ES03 software download page』(load_result.zip)で入手できます。このデータは、2000ユーザ用のデフォルトレイアウトで8時間に設定されているSSOログインおよびCUIC LDレポートを使用した2000 UCCE導入の例における、デスクトップおよびスーパーバイザの操作のプロキシの状態を表します。これを使用して、Nginxを同等のハードウェアで使用するインストールのコンピューティング、ディスク、ネットワーク要件を導き出すことができます。

トラブルシュート

SSO

1. デスクトップのリダイレクトがプロキシ経由ではない
    1. proxymap.txt、server_filterファイルなどのさまざまな設定で、ホスト名が実際のvmホスト名に従って正しいケースで設定されていることを確認します。

    2. CCE Web AdminからSSOに登録する際に同じ情報がコンポーネントにプッシュされるため、CCEインベントリでIdSが正しい大文字のホスト名で追加されていることを確認します。
2. SSOログインが行われない
    1. プロキシホストに対してIdS-IDP信頼が確立されていることを確認します。

SELinux

1. Nginxがデフォルトで起動していないか、Finesseエージェントデスクトップにアクセスできない場合は、次のコマンドでSELinuxをpermissiveモードに設定します。

```
setenforce 0
```

2. **systemctl restart nginx**コマンドを使用して、Nginxを再起動してみます。
3. 違反は/var/log/messagesおよび/var/log/audit/audit.logで確認できます。
4. 次のいずれかのコマンドを使用して、これらの違反に対処するための許可ルールを含む.teファイルを再生成する必要があります。

```
cat /var/log/audit/audit.log | audit2allow -m nginx1 > nginx1.te. # this will create nginx1.te fil
                or
ausearch -c 'nginx' --raw | audit2allow -M  my-nginx  # this will create my-nginx.te file
```

5. selinux-nginx-rhel-master-modified/nginxディレクトリにある元のnginx.teファイルを、新しく生成された許可ルールで更新します。
6. **make**コマンドを使用して同じものをコンパイルします。
7. nginx.ppファイルが再生成されます。
8. ポリシーをsemoduleコマンドでロードします。

```
semodule -i nginx.pp
```

9. 次のコマンドを使用して、SELinuxがenforceモードになるようにします。

```
setenforce
```

10. システムをリブートします。
11. 必要な違反が修正されるまで、この手順を繰り返します。

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。