# リッチクライアントのセキュリティ

# Magic xpa

本マニュアルに記載の内容は、将来予告なしに変更することがあります。これらの情報について MSE（Magic Software Enterprises Ltd.）および MSJ（Magic Software Japan K.K.）は、いかなる責任も負いません。

本マニュアルの内容につきましては、万全を期して作成していますが、万一誤りや不正確な記述があったとしても、MSE および MSJ はいかなる責任、債務も負いません。

MSE および MSJ は、この製品の商業価値や特定の用途に対する適合性の保証を含め、この製品に関する明示的、あるいは黙示的な保証は一切していません。

本マニュアルに記載のソフトウェアは、製品の使用許諾契約書に記載の条件に同意をされたライセンス所有者に対してのみ供給されるものです。同ライセンスの許可する条件のもとでのみ、使用または複製することが許されます。

当該ライセンスが特に許可している場合を除いては、いかなる媒体へも複製することはできません。ライセンス所有者自身の個人使用目的で行う場合を除き、MSE または MSJ の書面による事前の許可なしでは、いかなる条件下でも、本マニュアルのいかなる部分も、電子的、機械的、撮影、録音、その他のいかなる手段によっても、コピー、検索システムへの記憶、電送を行うことはできません。

サードパーティ各社商標の引用は、MSE および MSJ の製品に対するコンパチビリティに関しての情報提供のみを目的としてなされるものです。

本マニュアルにおいて、説明のためにサンプルとして引用されている会社名、製品名、住所、人物は、特に断り書きのないかぎり、すべて架空のものであり、実在のものについて言及するものではありません。

Magic は Magic Software Japan K.K. の登録商標です。
Magic xpa は Magic Software Enterprises Ltd. のイスラエルその他の国での商標または登録商標です。
Magic xpa Enterprise Studio、Magic xpa Enterprise Client、Magic xpa Enterprise Server および Magic xpa RIA Server は Magic Software Japan K.K. の商標です。
Pervasive.SQL® は Pervasive Software, Inc. の商標です。
IBM®, iSeries™, xSeries®, DB2® および WebSphere® は、IBM Corporation の商標または登録商標です。
Microsoft® および FrontPage® は、Microsoft Corporation の登録商標です。また、Windows™, WindowsNT™ および ActiveX ™ は Microsoft Corporation の商標です。
Oracle® は Oracle Corporation の登録商標です。
Linux® は Linus Torvalds の登録商標です。
GLOBEtrotter® と FLEXlm® は、Macrovision Corporation の登録商標です。
Interstage® は、富士通株式会社の登録商標です。
JBoss™ は、JBoss Inc. の商標です。
Systinet™ は、Hewlet-Packard Development Company の商標です。

一般に、会社名、製品名は各社の商標または登録商標です。
MSE および MSJ は、本製品の使用またはその使用によってもたらされる結果に関する保証や告知は一切していません。この製品のもたらす結果およびパフォーマンスに関する危険性は、すべてユーザが責任を負うものとします。

この製品を使用した結果、または使用不可能な結果生じた間接的、偶発的、副次的な損害（営利損失、業務中断、業務情報の損失などの損害も含む）に関し、事前に損害の可能性が勧告されていた場合であっても、MSE および MSJ、その管理者、役員、従業員、代理人は、いかなる場合にも一切責任を負いません。



2013 年 07 月 30 日

# 1 概要



# 2 サポートするアーキテクチャー



# 3 RIA のライフサイクル



# 4 Magic xpa RIA のビルトインのセキュリティ機能



# 5 ネットワークとホスト関連の脅威と対策

# 6 アプリケーション関連の脅威と対策



# 7 推薦事項

# 第 1 章　概要

このドキュメントは、インターネット関連のセキュリティ問題に直面しているリッチインターネット・アプリケーション（RIA）をサポートする基本的なアーキテクチャーについて解説するものです。

## サポートするアーキテクチャー

第 2 章は、RIA を含む Magic xpa のインターネット・アプリケーションがサポートするアーキテクチャーについて解説します。

## RIA のライフサイクル

第 3 章は、Magic xpa の RIA の一般的なライフサイクルについて扱っています。ここでは、ユーザがアプリケーションを操作する上で、どのようなタイプの情報がクライアントとサーバ間で送られるかについて説明しています。

## ビルトインのセキュリティ対策

第 4 章は、Magic xpa を非常に信頼性の高いプラットホームとするためのセキュリティ関連の機能について説明しています。

## セキュリティの脅威と対策

第 5 章と第 6 章は、既知のセキュリティに関する脅威について扱っています。Magic xpa の RIA を作成する際、インターネット・アプリケーションの脅威に対する対応について知っていることで、様々なセキュリティ上の問題を回避することができます。他の脅威については、推奨する対策を提示します。

このドキュメントで提示しているセキュリティ上の脅威は、さまざまな外部ソースから集めたものです。第 5 章および第 6 章の大部分の情報は、MSDN の「Web アプリケーション　セキュリティ強化：脅威とその対策」というタイトルの Microsoft のオンラインドキュメントを参考にしたものです。

あらゆるセキュリティ上の脅威についての Magic xpa での考え方は、必要に応じて提供されます。

## 推薦事項

Magic xpa のプラットホームは、本質的に、アプリケーションの実際のビジネス関連の機能に関してフォーカスするために、大部分のセキュリティ上の問題を処理して、開発者が扱わなくて良いようにします。しかし、アプリケーションのセキュリティレベルを高くするための機能や能力を持ち合わせています。第 7 章は、推奨された実装方法についての情報について説明しています。

# 第 2 章 サポートするアーキテクチャー

Magic xpa のアプリケーションサーバの実行環境は、Magic xpa によって提供される分散型アプリケーションアーキテクチャーのモジュールを使用して構成されます。Magic xpa のアプリケーションサーバは、大規模な並行ユーザを処理するために設計されており、自動的に最適化されたコンテキスト管理で、フロントエンドを提供します。

## 配信モジュール

Magic xpa RIA を構成する為に必要な基本モジュールは以下の通りです。

### RIA クライアント

Magic xpa は、Web ブラウザを使用しないクライアント・モジュールを提供します。このクライアント・モジュールは、Magic xpa RIA のユーザインタフェースとクライアント側のロジックに反映される汎用的なモジュールです。Magic xpa RIA クライアントは、クライアント側で自動的にインストールされる自動インストール可能なモジュールです。

RIA クライアントの Window 版の配備は、アプリケーション・ベンダによって署名されたクライアント・モジュールをインストールするために Microsoft の ClickOnce を使用します。ClickOnce は、以下のフォルダ内の RIA クライアントファイルをインストールします。

```
c:¥users¥username¥AppData¥Local¥Apps¥3.0¥obfuscatedfoldername¥obfuscatedfoldername¥appname
```

各 RIA クライアントファイルのために、ClickOnce は次の 2 つのファイルを作成します：

- .manifest file
- .cdf-ms file.

これらのファイルには、ClickOnce の自動アップデータがどのファイルを使用するかという、色々な dll に関する情報が含まれています。

モバイル RIA クライアントのモバイルへの配備は、アプリケーション・ベンダによって署名されるクライアント・モジュールをインストールすることによって行われます。

### WWW サービス機能

Web サーバは、リモートの RIA クライアントから HTTP リクエストを受け取ることを要求されます。Web サーバは、Magic xpa のインターネットリクエスタを使用して Magic xpa アプリケーションサーバに HTTP リクエストを送ります。

### Magic xpa のインターネットリクエスタ

Magic xpa は、Web サーバを利用するインターネットリクエスタのモジュールを提供します。RIA クライアントがリクエスタにリクエストを作成すると、モジュールは Magic xpa アプリケーションサーバに渡すデータを含んだリクエストを渡します。

インターネットリクエスタのモジュールは、MRB によって維持されるサーバエンジンと接続プールを使用して、利用できるアプリケーションサーバを位置づけることができます。

### MRB（Magic xpa Request Broker）

MRB は、利用できる全ての Magic xpa アプリケーションサーバのエンジンを処理して、インターネットリクエスタから利用できるアプリケーションサーバ・エンジンまで各リクエストを制御します。MRB は、様々なフェールオーバを処理するために、ロードバランシング機能とリカバリ機能を提供しています。

### Magic xpa アプリケーションサーバ

Magic xpa アプリケーションサーバは、リッチインターネット・アプリケーションの実行環境の中心に位置付けられます。各リクエストを処理して、受信する各タイプのリクエストに対応するサーバ側のロジックを実行する中心的な実行ユニットになります。Magic xpa アプリケーションサーバは MRB の位置を知っており、そこに接続されている必要があります。これによってインターネットリクエスタが利用できるようになります。

Magic xpa アプリケーションサーバのエンジンは、一つのエンジンプロセスを使用して複数のリクエストを処理するように設計されています。これは、サーバのマルチスレッディング機能を使用して実現しています。

## モジュールの配布

異なる OS を使用しているマシン環境の場合でも、上記のモジュールは同じマシン上にインストールすることも、異なるマシンに分配することもできます。

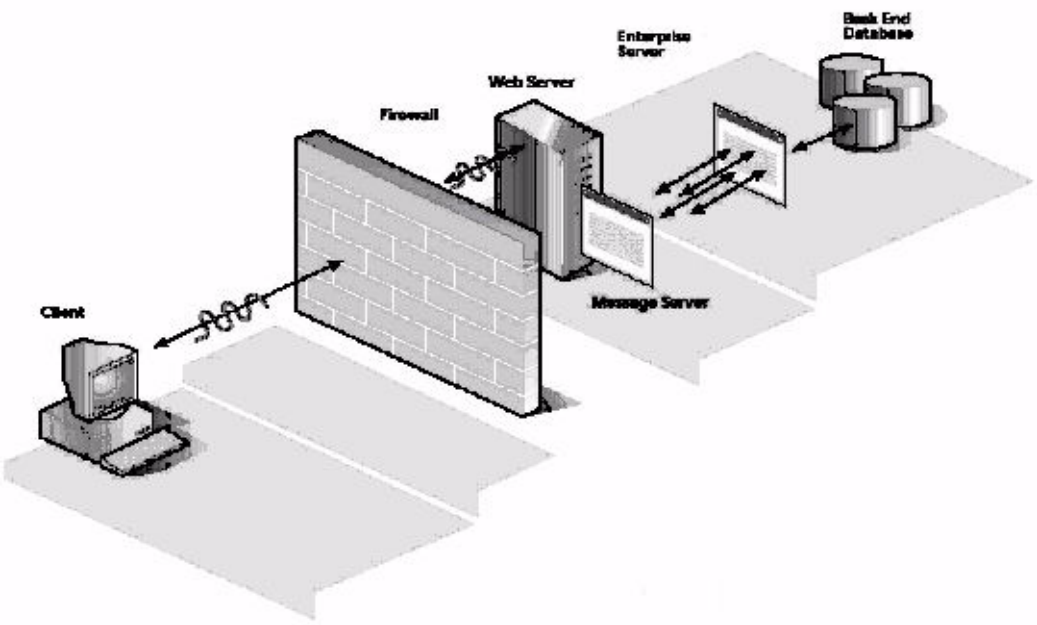


図 2-1：このイメージは、RIA クライアントがどのようにアプリケーションサーバと通信するか、そして、バックエンドが配信された Magic xpa モジュールとどのように相互作用するかを説明しています。

# 配信されたモジュールの間の通信

## リクエストのライフサイクル

RIA クライアントで送信されたあらゆるリクエストは、送られてから応答が返るまで定義済みのプロシージャを通して実行されます。利用内容や中身に関係なく、ライフスタイルは、各リクエストで同じになります。

## クライアントから Web サーバへ

一旦リクエストが送られると、HTTP またはセキュアな HTTP（HTTPS）で Web サーバに渡されます。一旦リクエストが Web サーバに渡されると、Web サーバ内で実行されるリクエスタはデータを取得します。

## リクエスタから MRB へ

リクエスタは、利用できる Magic xpa サーバエンジンのために、MRB に問い合わせを行います。MRB は、利用できるサーバエンジンのホスト名とポート番号をリクエスタに通知します。この通信は、TCP/IP で行われます。

## リクエスタから（Magic）サーバエンジン へ

一旦リクエスタが利用できるサーバエンジンの詳細情報を取得すると、サーバエンジンにリクエストの詳細を渡します。サーバエンジンはリクエストを処理して、結果をリクエスタに送り返します。この通信は、TCP/IP で行われます。

## Web サーバから RIA クライアントへ

一旦リクエスタが処理結果を受けとると、Web サーバによって応答が HTTP/HTTPS を使用して RIA クライアントへ送られます。

# 第 3 章 RIA のライフサイクル

あらゆるリッチインターネット・アプリケーションでは、通常、アプリケーション・フローを開始するために実行される最初のプログラムをエントリポイントとして定義しています。

## RIA タスクの初期化

RIA プログラムを実行するための最初のリクエストを受け取ると、サーバエンジンはコンテキストを開き、ユニークなコンテキスト ID を作成します。短い応答がクライアントに即時に返され、作成されたコンテキストを通知します。

コンテキストの応答としてクライアントでの短いプロセスを終了すると、2 番目のリクエストがアプリケーションサーバ向けに作成されます。その際、このコンテキストのために作成されたコンテキスト ID が使用されます。そして、クライアント（匿名、または、証明書の提供、パスワード入力で）を認証します。

初期化手順の 3 番目で最後のリクエストによって、RIA プログラムが読み込まれ、実行されます。

## サーバの応答

RIA プログラムを初期化の際、サーバの応答として、以下の要素を含む XML ベースの応答が返ります。

### プレゼンテーションレイヤ

RIA プログラムの呼び出しがタスクのプレゼンテーションレイヤの場合、クライアントが受け取る応答部分。この XML 部ページは、Maghic xpa Studio でプログラムのために定義された設計内容が含まれています。

### ロジックレイヤ

RIA プログラムの呼び出しがタスクのロジックレイヤの場合、クライアントが受け取る別の部分。この XML 部ページは、サーバ側の処理が必要となる場合、いつサーバエンジンに切り替えるかという指示と共に RIA プログラムとして定義されたクライアント側のロジックの全てが含まれています。

XML ベースの応答の中でプレゼンテーションとロジックの部分は、アプリケーションの処理中は固定で、サーバで作成されて、Web サーバと外部の環境からは表示されない状態で維持されます。

ロジックとプレゼンテーションの XML ファイルは、常にアプリケーションのクライアント部分として直接送られます。クライアントは、送られたファイルをキャッシュに格納します。アプリケーション定義が修正されると常に、変更された XML 要素がリクエストに応じてクライアントに再送信されます。

### データレイヤ

タスクのビュー定義にもとづいて、サーバからの応答に追加された XML 部は、タスクで必要とされる最初のデータを定義します。

## 処理中の操作

エンドユーザが RIA プログラムを操作している時、操作内容はサーバ側の処理（トランザクションのコミットや、サーバ側ロジックの実行など）を必要とするかもしれません。

## RIA クライアントからアプリケーションサーバへ

RIA クライアントは、サーバ側の動作毎に、第 2 章で説明したようなフローでアプリケーションサーバに HTTP リクエストを送ります。この種のリクエストは、変更されたデータと必要なサーバ側のロジックを XML フォーマットで送ります。

## アプリケーションサーバから RIA クライアントへ

アプリケーションサーバは XML を受け取り、クライアントに HTTP の応答を返します。HTTP の応答には、クライアント側で表示する新しいデータやクライアント側のロジックに対応する操作を指示するデータが含まれています。

# 第 4 章 Magic xpa RIA のビルトインのセキュリティ機能

Magic xpa のプラットホームには、少ない開発工数と配備作業でより厳しいセキュリティを実現する為に様々な仕組みが組み込まれています。

- HTTPS のサポート
- ベンダの信頼性認証
- 内部通信の暗号化
- スクランブル化された送信内容
- コードの難読化
- ストリーム化されたデータ
- 暗号化されたローカルキャッシュ

## HTTPS のサポート

Magic xpa プラットホームは、セキュアな HTTP レイヤ（HTTPS）上での Magic xpa RIA の実行を透過的にサポートします。Web サーバ側の設定が必要ですが、それ以外はアプリケーションとして対応する必要がありません。

## ベンダの信頼性認証

Windows クライアント用の RIA の実行には、Microsoft の ClickOnce 技術が使用されています。各 RIA で使用するマニフェストは、アプリケーションプロバイダによって取得された証明書によって作成され、署名されています。ClickOnce がアプリケーションを配備したり、アップデートしたりすると、サーバからダウンロードされた各ファイルのハッシュを計算して、ダウンロードされたアプリケーションのマニフェストに埋め込まれたものと比較します。

アプリケーションのマニフェストが署名され、アプリケーションファイルに対応するハッシュ値も変更されるため、修正することができません。ダウンロードされた後でアプリケーションファイルのハッシュが合わないと ClickOnce がアプリケーションを起動することを拒否するため、このファイルを改ざんすることはできません。

## 内部通信の暗号化

Magic リクエスタや MRB、Magic xpa エンジンの間の通信は、非対称型と対称型の暗号処理を使用して暗号化されます。暗号化のためにハードコートされたキーを使用していない為、対称型の暗号化メカニズムとキー長を定義することが可能です。

非対称の暗号化はハンドシェイク（1024 ビットのキー長をもつ RSA）のためにのみ使用されており、その点から対称型の暗号化が使用されます。アルゴリズムとキー長は、サポートされた暗号アルゴリズムとキー長に基づいて、開発者によって Magic xpa プラットホームの環境ファイルで設定することができます。

Magic xpa のモジュール間で接続を開く度に、それ自身の対称型のキーを持ちます。そして、それは（非対称の暗号化での）ハンドシェイクプロセス内で自動的にその場で作成されます。

## スクランブル化された送信内容

セキュアな HTTP レイヤに加え、Magic xpa は RIA クライアントと Magic xpa サーバ間に転送される実際の内容のセキュリティにもう一つのレベルを追加します。RIA クライアントと Magic xpa サーバエンジンの間のデータは、スクランブル化されます。スクランブル処理は、Magic xpa が持っているスクランブル・アルゴリズムを使用して行われます。

## コードの難読化

RIA クライアント・モジュールのコードは、通信中や認証時、クライアントの暗号化レイヤに流れる場合に、特にクライアントの処理の情報が漏れることを防止するために難読化されています。

## ストリーム化されたデータ

アプリケーション定義とアプリケーション・データに関連するすべての情報は、クライアントに直接送られます。この種の情報は Web サーバ上の共有フォルダには保存されません。また、Web サーバの背後で実行される Magic xpa サーバのプロセスのみで利用できます。

## 暗号化されたローカルキャッシュ

パフォーマンス向上のために、Magic xpa RIA クライアントは、サーバから送られるアプリケーションのメタデータをキャッシュに格納します。クライアントも、特定のセッションで使用されるデータビューのいくつかをキャッシュに格納します。ローカルキャッシュの開示や改ざんを防止するために、キャッシュに格納された情報は、56 ビットのキーを持つ DES によって暗号化されます。

# 第 5 章　ネットワークとホスト関連の脅威と対策

インターネット・アプリケーションは、悪意のある脅威の標的となります。

これらの脅威は、3 つの主要なカテゴリーに分類されます。

- ネットワークの脅威
- ホストの脅威
- アプリケーションの脅威

この章では、最初の 2 つのカテゴリーについて扱います。

| ノート |
| --- |
| この章の大部分の情報は、MSDN の "Web アプリケーションセキュリティ強化：脅威とその対策 " というタイトルの Microsoft のオンラインドキュメントによるものです。 |

## ネットワークの脅威

インターネット・アプリケーションを支えるネットワーク基盤は、以下の主要なコンポーネントから構成されています。

- ルータ
- ファイアウォール
- スイッチ

これらは、サーバとアプリケーションを攻撃と侵入から守る門番の働きをします。攻撃者は、設定が不十分なネットワーク・デバイスを利用する可能性があります。

ネットワークレベルでの脅威には以下のものがあります。

- 情報収集
- スニッフィング
- なりすまし
- セッション・ハイジャック
- サービス拒否

## 情報収集

ネットワーク・デバイスは、他のタイプのシステムと同様に検出してプロファイル（分析）できます。攻撃者は、通常、ポートスキャニングから始めます。開いているポートを確認すると、デバイスのタイプを見つけ、OS とアプリケーションのバージョンを決定するために、バナーグラブと列挙を使用します。この情報もとに武装することで、攻撃者はセキュリティパッチが更新されていない既知の脆弱性を攻撃することができます。

### 情報収集の防止策：

- （情報収集の）フットプリンティングのリクエストに応答しないように、ルータを設定してください。
- 未使用のプロトコルと不要なポートを無効にすることでフットプリンティングを防止するため、ネットワーク・ソフトウェア（例：ソフトウェア・ファイアウォール）を動かす OS を設定します。

### Magic xpa での対応

クライアントと Web サーバの間に、そして、Magic xpa コンポーネント（リクエスタや MRB、アプリケーションサーバ）の間にセキュアなメカニズムがある場合、上記の脅威はセキュアなメカニズムで防止できます。

セキュアなメカニズムは、SSL やクライアント側やサーバ側で実装される外部の暗号／解読メカニズムのどちらか（または両方）で実現しています。

## スニッフィング

" スニッフィング " または " 盗聴 " は、データ（例えば、平文のパスワードまたは設定情報）に対してネットワーク上でやり取りをモニタする行為です。単純なパケットスニファーを使用することで、攻撃者はすべての平文のデータを簡単に読むことができます。また、攻撃者は軽量のハッシュアルゴリズムを使用してパケットの暗号を解読したり、安全と考えていたデータ部分の解読を行ったりすることができます。パケットのスニフィングは、サーバ / クライアント間の通信経路に、パケットスニファーを設置する必要があります。

### スニッフィングの防止策：

- 強力な物理的セキュリティとネットワークの適切な分散化をおこなってください。これはトラフィックをローカルに集められることを防止する第一歩です。
- 認証証明書を含めて、通信内容は完全に暗号化してください。これは、攻撃者によってスニッフィングされたパケットに対する防御となります。SSL と IPSec（Internet Protocol Security）は、暗号化ソリューションの一 例です。

### Magic xpa での対応

Magic xpa RIA クライアントは、クライアントとサーバ間でセキュアな接続をサポートします。これは、クライアントから Web サーバに送られたり、応答として返されたりする実際のデータをモニタされることを防止します。ほとんどの場合、Web サーバやリクエスタ、MRB と Magic xpa サーバエンジンは、認可なくアクセスできないサブネット上の DMZ 内、またはその背後に配置されます。

## なりすまし

" なりすまし " は、ネットワーク上に正しい識別情報を隠蔽する手段です。なりすまし用の ID を作成するために、攻撃者はパケットの実際のアドレスではない偽のソースアドレスを使用します。なりすましは、攻撃用の発信元を隠蔽するため、ソースアドレスの規則に基づいてホストアクセスを制限するネットワークアクセス制御リスト（ACL）を回避するために使用されます。

巧みに作成された、なりすましパケットは発信元がトレースされることはありませんが、フィルタリング規則の組み合わせによって、なりすましパケットがネットワークから発信されることを防止することができます。そして、なりすましパケットをブロックすることができます。

### なりすましの防止策：

- 周囲の内部 IP アドレスから送信された受信パケットをフィルタリングしてください。
- 無効なローカル IP アドレスから発信されている送信パケットをフィルタリングしてください。

### Magic xpa での対応

ネットワークレベルのフィルタリングに加えて、Magic xpa RIA では、開発者が拡張されたセキュリティ対応のために追加されたクライアント側の識別を追加するようにアプリケーションを設計することができます。

クライアント PC と対話する RIA クライアント・モジュールの能力によって、Magic xpa サーバで照合や検証が行われるクライアントの組み込み情報を取り出すように開発することができます。Magic xpa の ClientGetUniqueMachineID 関数の利用は、特定のマシンがアプリケーションにアクセスすることや、アプリケーションのより機繊細な部分にアクセスする許可を与える 1 つの方法になります。

この ID は、次回インターネットへ接続した時点で変更されることに注意してください。

## セッション・ハイジャック

「中間者攻撃」として知られている、セッション・ハイジャックは、サーバまたはクライアントに対して、上流のホストが実際の正当なホストとであるかのように見せかけます。上流のホストが、ネットワークを操作している攻撃者のホストとなります。このため攻撃者のホストが正当な宛先であるように見せかけます。

### セッション・ハイジャックの防止策：

- 暗号化されたセッションネゴシエーションの使用してください。
- 暗号化された通信チャンネルの使用してください。

- TCP/IP の脆弱性（例えば予測可能なパケット・シーケンス）を修正するために、プラットホームのパッチに関する情報に常に気を止めてください。

### Magic xpa での対応

Magic xpa で特有の構文（通常の HTML ベースの送信と応答とは異なります ) を使用してセキュアなチャンネル（HTTPS）上で通信する Magic xpa の能力を利用することで、セッションハイジャッキングを不可能にすることができます。

## サービス拒否

サービス拒否は、サーバまたはサービスへの正当なユーザアクセスを拒否します。SYN フラッド攻撃は、ネットワークレベルのサービス拒否攻撃の一般的な例です。この方法は、簡単に実行することができ、トラッキングすることが困難です。攻撃の狙いは、処理能力を超えるより多くのリクエストをサーバに送信することです。攻撃は TCP/IP の接続確立メカニズムでの潜在的な脆弱性を利用して、サーバに未処理の接続キューを溢れさせます。

### サービス拒否の防止策：

- TCP 接続キューのサイズを増やしたり、接続の有効期間を短くしたり、接続キューが枯渇しないようにするために動的なバックログメカニズムを使用することができるようにレジストリを適切に設定して、TCP/IP スタックを堅固なものにしてください。
- ネットワーク侵入検知システム（IDS）を使用することで、SYN 攻撃を自動的に見つけ、対応することができます。
- 拡張された Web サーバの難読化を利用できるように、リバースプロキシを使用してください。

### Magic xpa での対応

この脅威は、アプリケーションまたは配備ツールとは関係ありません。それゆえ、ネットワークの設定だけで対応する必要があります。

## ホストの脅威

ホストに対する脅威は、アプリケーションが構築されるシステムソフトに向けられます。

ホストレベルの脅威には以下ものがあります。

- ウイルス、トロイの木馬やワーム
- フットプリンティング
- プロファイリング
- パスワードクラッキング
- サービス拒否
- 任意コード実行
- 不正アクセス

## ウイルス、トロイの木馬やワーム

“ウイルス”は、悪意のある動作を実行して、OS またはアプリケーションを破損させるように設計されたプログラムです。”トロイの木馬”は、悪意のあるコードが正常なデータファイルまたは実行可能プログラムのように見えること以外は、ウイルスに似ています。”ワーム”は、1 台のサーバから別のサーバに自己複製すること以外は、トロイの木馬と似ています。ワームは、確認できるファイルを定期的に作成しないため、検出が困難です。システムの動きが悪くなったり、他のプログラムの実行が停止したりするため、システムのリソースが消費し始めるようになってから気付くことがほとんどです。Code Red ワームは、IIS に影響を及ぼす悪名高いものの 1 つです。これは、特定の ISAPI フィルタ内のバッファオーバーフローの脆弱性に依存しています。

これらの 3 つの脅威は、実際の攻撃ですが、同時行われると、インターネット・アプリケーションや、これらのアプリケーションが実行されるホスト、これらのアプリケーションを配信するために使用されるネットワークに対する重大な脅威をもたらします。システムに対するこれらの攻撃が成功するには、多くの脆弱性（例えば、脆弱なデフォルト設定、ソフトウェアのバグ、ユーザの操作エラー、インターネットプロトコルに内在する脆弱性）が原因となります。

### ウイルス、トロイの木馬とワームの防止策：

- ファイアウォールとホストですべての不要なポートをブロックしてください。
- プロトコルとサービスを含む未使用の機能を無効にしてください。
- デフォルトの設定値を変更して強化してください。

### Magic xpa での対応

外部のセキュリティユーティリティ（例：マルウエアデテクタ）を利用することに加え、実行エンジンの環境設定を使用している Magic xpa のポート使用を設定することができます。

## フットプリンティング

フットプリンティングの例には、ポートスキャンと Ping スイープ、NetBIOS 列挙があります。これらは、より重要な攻撃に対する準備のために必要なシステムレベルの情報を収集するために、攻撃者によって使用されることがあります。フットプリンティングによって潜在的に明らかにされる情報のタイプには、アカウントの詳細や OS、他のソフトウェアバージョン、サーバ名、データベーススキーマの詳細が含まれています。

### フットプリンティングの防止策：

- 不要なプロトコルを無効にしてください。
- ファイアウォールを適切に設定しポートをロックしてください。
- 多重防御の為に TCP/IP と IPsec を使用してください。
- Web サーバを設定して、バナーグラブを経由した情報漏洩を防止してください。
- フットプリンティングのパターンを検出して、不審なトラフィックを拒否できるように IDS を使用してください。

### Magic xpa での対応

この威嚇は、アプリケーションまたは配備ツールには関係しません。このため、サーバ PC の設定だけが対象になります。

## パスワードクラッキング

攻撃者が匿名でサーバに接続できない場合、認証された接続を確立しようとします。そのためには、攻撃者は有効なユーザ名とパスワードの組合せを知っていなければなりません。デフォルトのアカウントを使用している場合、攻撃者に有利に開始させることになります。攻撃者はアカウントのパスワードを解析するだけだからです。空白または脆弱性のあるパスワードを使用すると、攻撃者の作業をより簡単にしてしまいます。

### パスワード解析の防止策：

- すべてのアカウントタイプで解析しにくいパスワードを使用してください。
- エンドユーザ・アカウントのロックアウト・ポリシーを適用して、パスワードを推測するために使用されるリトライ回数を制限してください。
- デフォルトのアカウント名を使用しないで、標準的なアカウント（例えば、多くのインターネット・アプリケーションで使用される Administrator のアカウントと匿名のインターネット利用アカウント）の名前を変更してください。
- パスワードのハッキングの試行パターンに対してログインが失敗するように監視するようにしてください。

### Magic xpa での対応

Magic xpa は、標準的なユーザ認証機能（例えば、Active Directory や LDAP）を利用することができます。しかし、この脅威は、アプリケーションまたは配備 - ツール関連した脅威でありません。このため、適当なユーザアカウント・ポリシーの設定によって対応する必要があります。

## サービス拒否

サービス拒否は、インフラ内の複数の標的を狙う多くの方法によって行われます。ホストでは、攻撃者はアプリケーションに対して強引にサービスを中断させることができます。また、攻撃者はアプリケーションが実行しているサービス、または、サーバを実行する OS に存在する脆弱性を知っている可能性があります。

### サービス拒否の防止策：

- サービス拒否を念頭に置いてアプリケーションやサービス、OS を設定してください。
- サービス拒否に対して TCP/IP スタックを強化してください。
- アカウントロックアウト・ポリシーを悪用して WKS(Well Known Service) アカウントをロックアウトできないようにしてください。
- アプリケーションが高負荷のトラフィックを処理することができるようにして、異常に高い読み込みを処理できるようにしきい値を設定してください。
- アプリケーションのフェールオーバ機能を見直してください。
- 潜在的なサービス拒否攻撃を見つけることができる IDS を使用してください。

### Magic xpa での対応

Magic xpa のリクエストフィルタリングで、各タイプのリクエストに割り当ててられたサービスレベルを制御することができます。さらに、Magic xpa プラットホームのスケーラブルな機能は、サービス需要内で異常に増大する可能性に対応することができます。それでも、大部分の脅威は、ネットワークとサーバマシンの設定で対応する必要があります。

## 任意コードの実行

攻撃者がサーバで悪意のあるコードを実行することができる場合、攻撃者はサーバのリソースを漏洩したり、下流システムへの更なる攻撃を開始したりすることができます。サーバが、攻撃者のコードの実行に過度な権利が与えられている場合、任意のコード実行によってもたらされるリスクが増加します。一般的な脆弱性には、IIS の設定の脆弱性やパストラバーサルとバッファオーバーフロー攻撃を可能にするパッチが当てられていないサーバを含んでいます。そして、その両方は任意コード実行を可能にすることになります。

### 任意コード実行の防止策：

- ”./”が付加された URL を拒絶して、パストラバーサルを防止するように Web サーバを設定してください。
- ACL を使用してシステムコマンドとユーティリティをロックしてください。

### Magic xpa での対応

Magic xpa RIA は、クライアントマシンから利用可能となると、一般的な HTTP ポートのみ必要となります。さらに、この脅威はアプリケーションや、配備ツールには関係しないため、上記の対策に従われなければなりません。それゆえに、ホスト側の設定で対応しなければなりません。

## 不正アクセス

アクセス制御が不十分な場合、無許可のユーザが制限された情報をアクセスしたり、制限された処理を実行させたりすることになります。一般的な脆弱性は、Web の権利設定と脆弱な NTFS の権利設定を含む、脆弱な Web 接続コントロールが含まれています。

### 不正アクセスの防止策：

- セキュアな Web の権利設定を行ってください。
- 制限された NTFS の権利設定によってファイルとフォルダを保護してください。

### Magic xpa での対応

この脅威はアプリケーションや、配備ツールには関係しないため、ホスト側の設定で対応しなければなりません。

# 第 6 章 アプリケーション関連の脅威と対策

認証（データと一般のデータ管理の暗号化）に関するアプリケーション設計は、インターネット・アプリケーションをセキュアなものにすることで重要な役割を担います。

**ノート**

この章の大部分の情報は、MSDN の "Web アプリケーションセキュリティ強化：脅威とその対策 " というタイトルの Microsoft のオンラインドキュメントによるものです。

アプリケーション - レベルの脅威を分析する良い方法は、アプリケーションの脆弱性をカテゴリーごとに整理することです。この章の以降の節とこのドキュメントの至る所に記述しているいくつかのカテゴリーは、アプリケーションに対する主要な脅威を集めており、以下のテーブルでまとめています。

表 6-1 アプリケーションの脆弱性カテゴリーによる脅威

| カテゴリ | 脅威 |
|---|---|
| 入力検証 | バッファオーバーフロー<br>クロスサイトスクリプティング<br>SQL インジェクション<br>正規化 |
| 認証 | ネットワーク盗聴<br>ブルートフォース攻撃<br>辞書攻撃<br>Cookie リプレイ攻撃<br>資格情報の盗用 |
| 承認 | 権限の昇格<br>機密データの漏洩<br>データの改ざん |
| 構成管理 | 管理インターフェースへの不正アクセス<br>構成ストアへの不正アクセス<br>テキスト形式の構成機密情報の取得<br>個人記録の不足<br>過度な権利が与えられたプロセスとサービスアカント |
| セッション管理 | セッションハイジャッキング<br>セッション・リプレイ<br>中間者 |
| 暗号化 | 脆弱なキーの作成／管理<br>脆弱またはカスタム暗号化<br>チェックサム のなりすまし |
| パラメータ改ざん | クエリ文字列の操作<br>フォーム・フィールドの操作<br>Cookie の操作<br>HTTP ヘッダの操作 |
| 例外管理 | 攻撃者による実装情報の開示<br>サービス拒否 |
| 監査とロギング | ユーザによる操作拒否<br>攻撃者がトレースされずにアプリケーションを利用する<br>攻撃者がトレースを隠す |

## 入力検証

アプリケーションが入力データのタイプや長さ、フォーマットまたは範囲について根拠がないことを攻撃者が見つけると、入力検証はセキュリティ問題となります。攻撃者は、アプリケーションを脆弱にできるように巧みに作成された入力データを送ることができます。

ネットワークとホストレベルのエントリポイントが完全に保護されている場合、アプリケーションが公開する公開インターフェースが唯一の攻撃対象となります。アプリケーションへの入力は、システムをテストする方法とも、攻撃者に代わってコードを実行する方法ともなります。アプリケーションは、以下の攻撃に影響される可能性があります。

- バッファオーバーフロー
- クロスサイトスクリプト
- SQL インジェクション
- 正規化

## バッファオーバーフロー

バッファオーバーフローによる脆弱性は、サービス拒否攻撃またはコードインジェクションへと導くことができます。サービス拒否攻撃は、プロセスの異常終了を引き起こします。コードインジェクションは、攻撃者から送られたコードを実行することで、プログラムの実行アドレスが変更されます。以下のコードは、バッファオーバーフローの一般的な例を示しています。

```
void SomeFunction( char *pszInput ){
char szBuffer[10];
// 型チェックが行われなければ、入力内容がバッファに直接コピーされます。
strcpy(szBuffer, pszInput);
. . .
}
```

### バッファオーバーフローの防止策：

- 厳密な入力検証を実行してください。これは、バッファオーバーフローに対する防御の第一歩です。入力内容がコンテナの限界を超えること許可するようなバグがアプリケーションに存在する場合、予想外の入力処理が脆弱性の要因となります。タイプや長さ、フォーマット、範囲を確認することによって、入力を制限してください。

### Magic xpa での対応

Magic xpa アプリケーションの開発者は、基本的なデータタイプのオーバフローを考慮する必要はありません。開発者は簡略化されたデータタイプを処理し、Magic xpa のエンジンは事前に対応可能なあらゆるバッファオーバーフローを防止します。

## クロスサイトスクリプティング (XSS)

Web ブラウザが信頼された Web サイトに接続している間、XSS 攻撃によってユーザの Web ブラウザで任意のコードを実行させることができます。攻撃はアプリケーションのユーザを標的としており、アプリケーションそのものではありません。しかし、攻撃の媒体としてアプリケーションを使用します。

スクリプトコードが Web ブラウザによって信頼されたサイトからダウンロードされるため、Web ブラウザはコードが正当かどうかを確認することができません。Internet Explorer のセキュリティゾーンは、保護されていません。攻撃者のコードが信頼されたサイトに関連する Cookie にアクセスし、ユーザのローカル PC に保存されるため、ユーザの認証 Cookie は典型的な攻撃対象となります。

### クロスサイトスクリプティングの例

攻撃を始めるために、攻撃者は巧みに作られたハイパーリンク（たとえば、ユーザに送られた電子メールにハイパーリンクを埋め込んだり、ニュースグループに悪意のあるリンクを追加したりして）をクリックするようにユーザに信じ込ませる必要があります。このリンクは、アプリケーション内の脆弱なページをポイントし、HTML の出力ストリームによって非認証入力を返します。たとえば、以下の 2 つのリンクを考えます。

本物のリンクは、以下の通りです。

```
www.yourwebapplication.com/logon.aspx?username=bob
```

悪意のあるリンクは、以下の通りです。

```
www.yourwebapplication.com/logon.aspx?username=<script>alert('hackercode')</script>
```

アプリケーションがクエリ文字列を受け取り、適切に検証しないで Web ブラウザに返すと、スクリプトコードは Web ブラウザで実行されます。上記の例は、無害なポップアップ・メッセージを表示します。正しいスクリプトを使用すると、攻撃者は簡単にユーザの認証 Cookie を取得することができ、Web サイトにそれを送り、認証されたユーザとして標的となる Web サイトに向けたリクエストを作成することができます。

### Magic xpa での対応

Web ブラウザを使用しないため、RIA クライアントのモジュールは、コードまたはスクリプトによるクライアント側の操作ができません。" ソースの表示 " オプションがなく、エンドユーザによってスクリプトを追加することはできません。

## SQL インジェクション

SQL インジェクション攻撃は、入力認証の脆弱性を利用して、データベースで任意のコマンドを実行するものです。アプリケーションが入力内容を使用してダイナミックな SQL ステートメントを作成してデータベースにアクセスする場合に発生する可能性があります。フィルタリングされてないユーザ入力を含む文字列を受け取るストアドプロシージャーがコードとして含まれている場合も発生する可能性があります。SQL インジェクション攻撃を使用することで、攻撃者はデータベースで任意のコマンドを実行させることができます。アプリケーションがデータベースに接続する過剰な権限が与えられたアカウントを使用すると、問題が深刻化します。この場合、データの読み出し、操作、破壊が可能であるだけでなく、OS のコマンドを実行して他のサーバを侵害するためにデータベース・サーバを利用することができます。

### SQL インジェクションの例

検証されないユーザ入力をデータベースクエリに組み込んだ場合、アプリケーションは SQL インジェクション攻撃に影響する場合があります。フィルタリングされないユーザ入力でダイナミックな SQL ステートメントを作成するコードは、特に影響が大きくなります。以下のコードを参考にしてください。

```
SELECT * FROM Users
WHERE UserName ='" + txtuid + "'
```

攻撃者は、シングルクォーテーションとセミコロンを追加してからステートメントを記述することで SQL を差し込み、意図したコマンドを実行させることができます。txtuid フィールドに以下の文字列を入力したと想定します。

```
'; DROP TABLE Customers -
```

この結果は、以下のようなステートメントがデータベースに送られ実行されます。

```
SELECT * FROM Users WHERE UserName='''; DROP TABLE Customers --'
```

アプリケーションのログインがデータベースに対して十分な許可が与えられていた場合、これによって Customers テーブルは削除されます。ダブルダッシュ（--）は、SQL コメントを意味して、Trailing クオートのようにプログラマによって追加された他の文字をコメントアウトするために使用されます。

他にもより巧妙なトリックを実行させることができます。txtuid フィールドに以下の内容を入力します。

```
' OR 1=1 -
```

これにより、以下のコマンドが作成されます。

```
SELECT * FROM Users WHERE UserName='' OR 1=1
```

1=1 は常に真になるため、攻撃者は Users テーブルから全ての行のデータを読み出すことができます。

### SQL インジェクションの防止策：

- 厳密な入力検証を実行してください。アプリケーションは、リクエストをデータベースに送る前に、その入力を検証する必要があります。
- 入力文字列が実行可能なステートメントとみなされないようにするため、データベース・アクセス にはパラメータ化されたストアドプロシージャーを使用してください。ストアドプロシージャーを使用しない場合、SQL コマンドを作成する際には、SQL パラメータを使用してください。
- データベースに接続するには、権利レベルの低いアカウントを使用してください。

### Magic xpa での対応

Magic xpa RIA の配備において、プログラマがダイレクト SQL ステートメントの一部としてこれらの文字列を故意に使用しない限り、入力文字列は実行可能なステートメントとは見なされません。さらに、英数字のデータに基づく SQL の範囲や位置付けのプロシージャは、シングルクォーテーションと一緒にデータベース・サーバに渡されます。それによって SQL インジェクションの可能性を回避できます。

## 正規化

同じ標準名（正規名）を解決する形式の異なる入力は、正規化として表されます。それが入力としてプログラムに渡されるリソース名に基づいてセキュリティが決定される場合、コードは特に正規化の問題に影響されやすくなります。ファイルやパス、URL は、各ケースで同じ名前を意味するために多くの異なる方法を使用するため、正規化に対して脆弱性のあるリソースタイプになります。ファイル名も、問題を含んでいます。たとえば、一つのファイルは、以下のように表現できます。

```
c:¥temp¥somefile.dat
somefile.dat
c:¥temp¥subdir¥..¥somefile.dat
c:¥ temp¥ somefile.dat
..¥somefile.dat
```

理想的には、コードはファイル名の入力を受け取るべきではありません。このような場合、名前は（指定されたファイルのアクセスを認めるかどうかというような）セキュリティ上の決定を行う前に正規表現に変換しなければなりません。

### 正規化問題の防止策：

- 可能であれば、ファイル名を避け、代わりにエンドユーザによって変更できない絶対パスを使用するようにしてください。
- ファイル名を入力として受け取る必要がある場合は、ファイル名が適切な書式であり、アプリケーションのコンテキストの範囲内で検証してください。たとえば、アプリケーションのディレクトリ階層内であることを確認してください。
- 文字のエンコーディングが正しく、入力内容の表示方法が制限されることを確認してください。

### Magic xpa での対応

サーバ側の相対パスがクライアント側のフルパスとして関係があることを維持するには、Magic xpa の論理名を利用することを推奨します。

## 認証

要求条件にもとづいて、選択可能な認証メカニズムがいくつかあります。それらが正しく選択されず、実装されないと、認証メカニズムは攻撃者がシステムにアクセスができるような脆弱性をさらけ出すことになります。

認証の脆弱性を利用する脅威には、以下のものがあります。

- ネットワーク盗聴
- ブルート フォース攻撃
- 辞書攻撃
- Cookie リプレイ攻撃
- 資格情報の盗用

## ネットワーク盗聴

認証用の証明書がクライアントからサーバに平文で送られると、同じネットワーク上のホストで簡単なネットワークのモニタソフトアを使用することで、攻撃者は通信内容を傍受することができ、ユーザ名とパスワードを取得することができます。

### ネットワーク盗聴の防止策：

- ネットワーク上にパスワードを送らない認証メカニズム（例えば Kerberos プロトコルまたは Windows 認証）を使用してください。
- ネットワーク上にパスワードを送らなければならない場合は、パスワードが暗号化されていることを確認してください。または、SSL のような暗号化された通信チャンネルを使用してください。

### Magic xpa での対応

Magic xpa RIA クライアントは、クライアントとサーバの間でセキュアな接続をサポートします。これは、クライアントと Web サーバの間でやり取りされる実際のデータがモニタされることを防止することができます。ほとんどの場合、Web サーバとリクエスタ、MRB、Magic xpa のサーバエンジンは、認証がないとアクセスできないサブネット内の非武装地帯の中や背後に配置されます。さらに、クライアントからサーバに送られる Magic xpa に認証証明書はコンテキスト ID とともに暗号化されて、スクランブルをかけられます。そして、ネットワーク盗聴ならびにリプレイ攻撃を防止します。

## ブルート フォース攻撃

ブルート フォース攻撃は、ハッシュ化されたパスワードやハッシュ化されたり暗号化されたりした他の機密情報を PC の性能に依存して解読するものです。危険を減らすには、強力なパスワードを使用してください。

## 辞書攻撃

この攻撃は、パスワードを取得するために使用されます。大部分のパスワードシステムは、平文のパスワードまたは暗号化されたパスワードを保存しません。信用できないキーは、データストア内のすべてのパスワードの漏洩につながるため、暗号化パスワードの保存を避けています。キーが失われると、すべてのパスワードが無効になることを意味します。

大部分のユーザストアの実装は、パスワードのハッシュまたはダイジェストを保持します。ユーザが指定したパスワードの値に基づくハッシュを再計算して、データベースに保存されたハッシュ値と比較することによってユーザ認証が行われます。攻撃者がハッシュ化されたパスワードのリストを取得すると、パスワードのハッシュ値を解読するためにブルート フォース攻撃を使用することができます。

辞書攻撃では、攻撃者は辞書（または異なる言語の複数の辞書）内の全ての単語を繰り返し利用するプログラムを使用して、各単語に対するにハッシュ値を計算します。結果として生じるハッシュ値は、データストア内の値と比較されます。

### 辞書攻撃の防止策：

- 複雑で不規則な単語を含み、大文字、小文字、数字や特殊文字が含まれている強力なパスワードを使用するようにしてください。
- ユーザストア内に非可逆性のパスワードハッシュ値を保存してください。また、パスワードハッシュを持つ（暗号化された強力な乱数の）文字列を組み合わせてください。

### Magic xpa での対応

Magic xpa は、簡単で強力な認証機能（例えば、LDAP や Active Directory）を利用することができます。

## Cookie リプレイ攻撃

この種の攻撃を使用することで、攻撃者はモニタリング・ソフトウェアを使用してユーザの認証 Cookie を取得し、偽の識別環境下でアプリケーションをアクセスします。

### Cookie リプレイの防止策：

- 認証 Cookie を送る場合は、常に SSL によって提供される暗号化された通信チャンネルを使用してください。
- Cookie のタイムアウト値を比較的短い時間になるように設定してください。これでもリプレイ攻撃を防げない場合がありますが、セッションがタイムアウトになることで、攻撃者が再認証されることなくリクエストを再送信できる時間間隔を短くしてください。

### Magic xpa での対応

Magic xpa RIA は Web ブラウザを使用しないため、いかなる Cookie 関連のサポートも必要としません。セッションとコンテキスト管理は、RIA クライアントによって内部的に実行されます。

## 資格情報の盗用

アプリケーションがユーザのアカウント名とパスワードを格納する独自のユーザストアを実装している場合、プラットホーム（たとえば、Microsoft の Active Directory® サービス、または、Security Account Manager のユーザストア）によって提供される資格情報のストアとセキュリティを比較します。Web ブラウザの履歴やキャッシュも、将来の使用のためにユーザログイン情報が保存されます。ターミナルがログオンしたユーザ以外の誰かによってアクセスされ、同じページがアクセスされると、保存されたログイン情報が利用できます。

### 証明書窃盗の防止策：

- 強力なパスワードを使用するようにしてください。
- Salt（パスワード文字列に任意の文字列を付与してハッシュ化する手法）が追加された一方向のハッシュ値の形式でパスワード・ベリファイヤを保存してください。
- リトライ回数の上限値を設定し、エンドユーザに対してアカウントをロックアウトできるようにしてください。
- Web ブラウザのキャッシュによるログインアクセスの可能性に対処するために、ユーザが資格情報を保存しないことを許可したり、デフォルトポリシーとして強制したりする機能を作成してください。

### Magic xpa での対応

Magic xpa RIA は、Web ブラウザを使用しないため、クライアント側のいかなるロギングまたは過去のセッション履歴も作成されません。

## 承認

ユーザ ID と権利設定に基づいて、特定のリソースまたはサービスを利用可能または不可を設定します。

承認機能の脆弱性を利用した脅威には、以下のものがあります。

- 権利の昇格
- 機密データの漏洩
- データの改ざん
- おとり攻撃

## 権利の昇格

承認モデルを設計する際、権利を強力なアカウント（例えば、ローカルの管理者グループまたはローカルのシステムアカウントのメンバー）まで昇格させようとする攻撃者の脅威を考慮しなければなりません。この攻撃によって、攻撃者はアプリケーションとローカル PC を完全に制御することができます。

権利の昇格を防止するために利用できる主要な対策は、最小限の権利を持つプロセスやサービスとユーザアカウントを使用することです。

### Magic xpa の考え方

Magic xpa は、アプリケーションで様々な動作を管理するために、ユーザに対する機能割り当てや、グループ割り当てを柔軟にサポートするようにしています。権利を持つユーザに対する各機能を利用可能にするために、ユーザの権利割り当て機能を利用してください。

# 機密データの漏洩

機密を扱うデータが認証されていないユーザによって参照できる場合、機密データの漏洩が発生する可能性があります。機密データには、アプリケーション特有のデータ、例えば、クレジットカード番号や従業員の詳細、財政データ、アプリケーション特有のデータ（例：サービスアカウント証明書やデータベース接続文字列）が含まれます。機密データの漏洩を防止するには、データベースと設定ファイルのような固定ストア内やネットワーク上での通信内容を保護する必要があります。認証されて認可されたユーザのみが、指定されたデータにアクセスすることができるようにしなければなりません。システムレベルの設定データへのアクセスは、管理者に制限されなければなりません。

## 機密データの漏洩の防止策：

- 機密に関わるデータを潜在的に漏洩することができる処理へのアクセスを許可する前に、実行権の確認を行ってください。
- セキュアな Windows リソースに対して、強力な ACL を使用してください。
- 設定ファイルとデータベース内で機密を扱うデータを保存するために、標準的な暗号処理を使用してください。

## Magic xpa での対応

Magic xpa は、ユーザの権利設定機能やデータの暗号化、強力な認証機能とのシームレスな統合機能をサポートし、データの保護を可能にすることができます。

# データの改ざん

データの改ざんは、データが許可なく変更されることを意味しています。

## データの改ざんの防止策：

- 認可されたユーザだけがデータにアクセスし修正することができるようにするために固定ストア内でデータを保護し、強力なアクセス制御を使用してください。
- データを参照できるユーザと修正することができるユーザを区別するために、権利設定による保護を使用してください。

## Magic xpa での対応

ユーザ権利設定やデータ暗号化、強力な認証機能とのシームレスに統合された Magic xpa の内部機能によって、データを保護することができます。さらに、Web ブラウザを使用しないため、ブラウザベースのアプリケーションで可能な RIA の「ソース」を参照することはできません。

# おとり攻撃 ( フィッシング )

わずかな権利を持つエンティティがより多くの権利処理を実行することができるエンティティを入手することで、おとり攻撃が発生します。

この脅威に対応するには、適切な認証処理によって信頼されたコードでアクセスを制限するようにしなければなりません。このためには、.NET フレームワークのコードを使用して、保護されたリソースにアクセスするか、権利が必要な処理が実行されるときは、常に呼び出し元のコードを認証するように設定します。

## Magic xpa での対応

Magic xpa RIA のクライアントモジュールは、認証された証明書を使用して署名されたモジュールとして作成されています。アプリケーションがインストールされるときは常に、エンドユーザに対して署名内容が表示されます。アプリケーションのソースを検証した後でのみ、ユーザは自動的にクライアントモジュールのインストール処理を開始します。

# 構成管理

多くのアプリケーションは、構成管理インターフェースや機能をサポートしており、オペレータや管理者に構成パラメータの変更や Web サイトの内容の更新、定期的な保守作業を許可しています。

設定管理に関する脅威には以下のものがあります。

- 管理インターフェースへの不正アクセス
- 構成ストアへの不正アクセス
- 平文の構成内容の取得
- 個人記録の不足
- 過剰な権利が与えられたプロセスとサービスアカウント

## 管理インターフェースへの不正アクセス

管理インターフェースは、時々、アドミニストレータやオペレータ、コンテンツ開発者がサイトの内容と設定を管理することができるような、Web ページや個別のインターネット・アプリケーションを追加して提供される場合があります。これらのような管理インターフェースは、制限され認可されたユーザだけが利用できなければなりません。設定管理機能にアクセスすることができる悪意のあるユーザは、Web サイトを破壊したり、下流のシステムやデータベースにアクセスしたり、設定データを改ざんすることでアプリケーションを完全に動作不能にさせる可能性があります。

### 管理インターフェースへの不正アクセスの防止策：

- 管理インターフェースの数を最小限にしてください。
- 証明書を使用するなどして、強力な認証機能を使用してください。
- 複数のゲートキーパーを使用した強力な承認機能を使用してください。
- ローカルの管理者のみをサポートするように考慮してください。管理インターフェースに渡されるデータは細心の注意を払うべき性質のため、リモート管理が必要な場合は、（例えば、VPN 技術や SSL を使用した）暗号化されたチャンネルを使用して下さい。また、さらに危険を減らすために、内部ネットワーク上でリモート管理を行うコンピュータを制限するために IPsec ポリシーを使用することを考慮してください。

## 構成ストアへの不正アクセス

構成ストアに保存されるデータは機密が高いため、保存内容が適切に保護される必要があります。

### 設定ストアへの不正アクセスの防止策：

- テキストベースの構成ファイル（例えば Machine.config や Web.config）は、制限された ACL を構成してください。
- Web 空間の外部でカスタム構成ストアを置いてください。これによって、脆弱性を悪用するために Web サーバの構成情報がダウンロードされる可能性を防止します。

## 平文の設定内容の抜き取り

構成ストアへのアクセスを制限することは必須です。重要な多重防御メカニズムとして、機密に関わるデータ（例えばパスワードや接続文字列）を暗号化する必要があります。これによって、外部の攻撃者が機密に関わる構成データを取得することから防御することができます。また、悪意のある管理者や内部の従業員が機密に関わる詳細情報（例えば、他のシステムにアクセスが可能になるデータベースへの接続文字列とアカウント資格情報）を取得することを防止します。

## 個人記録の不足

監査や構成情報の変更ロギングの不十分な場合は、いつ、誰によって変更が行われたかを確認することができなくなります。オペレータによる誤操作や悪意のある変更によって構成が破壊され、アクセス権が与えられるような場合、まず変更を修正する必要があります。次に、同じ方法で破壊的な変更が行われないように、予防策を適用してください。監査とロギングは共有アカウントによって回避できることを覚えておいてください。これは、管理者およびユーザ / アプリケーション / サービスアカウントの両方に適用されます。管理者アカウントは、共有しないようにしてください。ユーザ、アプリケーション、サービスアカウントは、アカウントを使用してアクセス可能な 1 つのソースを識別でき、アカウントが与えられた権利に対する損害のみになるようなレベルで割り当てられなければなりません。

## 過剰な権利が与えられたプロセスとサービスアカウント

アプリケーションとサービスアカウントにシステムに関する構成情報へのアクセス権が与えられた場合、それらは攻撃者によって操作される可能性があります。この脅威の危険性は、最小限の権利サービスとアプリケーションアカウントを使用するポリシーを選択することで減らすことができます。設計上、明確に必要でない限り、アカウントに構成情報を修正する権利を与えないように気をつけてください。

# セッション管理

インターネット・アプリケーションのセッション管理は、アプリケーション・レイヤで行われます。セッションの安全対策は、アプリケーション全体の安全にきわめて重要です。

セッション管理に関する脅威には、以下ものがあります。

- セッションハイジャック
- セッションリプレイ
- 中間者攻撃

## セッションハイジャック

攻撃者がアプリケーションでユーザのセッションを表すために使用される認証トークン（主に Cookie）を獲得するためにネットワークモニタソフトウエアを使用すると、セッションハイジャック攻撃が発生します。取得された Cookie を使用することで、攻撃者はユーザのセッションを真似することが可能になり、アプリケーションにアクセスすることができます。攻撃者は、本物のユーザと同じ権利レベルを持つことになります。

### セッションハイジャックの防止策：

- SSL を使用して保護された通信チャンネルを作成して、HTTPS 接続でのみ認証 Cookie を渡すようにしてください。
- 別のセッションが開始されると、ユーザ認証が必要になるように、ログアウト機能を実装してセッションを終了させるようにしてください。
- SSL を使用しない場合、セッション Cookie 上で有効期間を設定してください。これにでセッションハイジャックが防止できる訳ではありませんが、攻撃者が利用できる時間を減らすことができます。

### Magic xpa での対応

オプションの SSL に加えて、Magic xpa は RIA クライアントがやり取りするメッセージの実際の内容が漏洩される可能性を減らすために、独自のスクランブル技術を使用しています。さらに、Magic xpa サーバとクライアント・モジュールは、セッション中の堅固なトレースと有効性のチェックを維持します。Magic xpa RIA クライアントの追加インスタンスは、別のクライアント・インスタンスによって使用中になっているセッションを使用することはできません。

## セッションリプレイ

ユーザのセッション・トークンが認証メカニズムを回避した攻撃者によって横取りされて、送信された場合、セッションリプレイが発生します。たとえば、セッショントークンが Cookie または URL 内で平文になっていると、攻撃者はそれを傍受することができます。次に、攻撃者は、乗っ取られたセッショントークンを使用してリクエストを送信します。

### セッションリプレイの脅威の防止策：

- 重要な機能を実行する際は再認証してください。たとえば、銀行取引アプリケーションで送金する前に、ユーザに対して再度アカウント／パスワードを入力させるようにしてください。
- すべての Cookie とセッショントークンを含めて、セッションに対する適切な期限設定を行ってください。
- データを保存しないオプションを作成し、セッションデータをクライアントに保存しないようにします。

### Magic xpa での対応

Magic xpa の RIA のコンテキスト管理機能は、コンテキスト管理の手段として Cookie を使用していません。このため、一旦セッションが終了すると、アプリケーションのフローや状態を「覚えている」クライアント側のトレースは残りません。さらに、クライアントからサーバに送られる Magic xpa の証明書はコンテキスト ID とともに暗号化されて、スクランブル化されています。そして、ネットワーク盗聴とリプレイアタックを防ぎます。

## 中間者攻撃

攻撃者が、送信者と指定された受取人の間に送られるメッセージを横取りすることで、中間者攻撃が発生します。次に、攻撃者は、メッセージを改ざんして最初の受取人に送り返します。受取人は、メッセージを受け取り、相手から来たものと見なして行動します。受取人がメッセージを送信者に送り返すと、攻撃者はそれを横取りし、改ざんして送信者に返します。送信者と受取人は、攻撃されたことに気付きません。

クライアント／サーバ間の通信を含む様々なネットワークリクエストには、Web リクエストや分散コンポーネントオブジェクトモデル（DCOM）が含まれており、リモート・コンポーネントや Web サービスの呼び出しを行います。これは中間者者攻撃の標的となる可能性があります。.

### 介入者攻撃の防止策：

- 暗号化を使用してください。データを送信する前に暗号化することで攻撃者は横取りすることはできますが、読み取ることができず改ざんすることもできません。攻撃者が読み取ることができなければ、改ざんする場所も分かりません。攻撃者がやみくもに暗号化されたメッセージを修正すると、最初の受取人はそれを解読することができず、その結果、それが改ざんされたということが分かります。
- HMAC（Hashed Message Authentication Codes）を使用してください。攻撃者がメッセージを改ざんすると、受取人の HMAC の再計算は失敗します。そして、データは無効となり拒否されます。

### Magic xpa での対応

Magic xpa は、HTTP のセキュアなレイヤを利用する能力に加えて、暗号化された内部の通信方法を提供します。Magic xpa の配布モジュール間の全てのメッセージは、暗号化されます。この暗号化は、内部の攻撃者が Magic xpa RIA のメッセージ内容を操作することを防止します。

## 暗号化

大部分のアプリケーションは、データを保護し、それが漏洩されず変更もされないことを保証するために、暗号化を使用しています。

暗号化されたアプリケーションの使用に関する脅威には以下のものがあります：

- 脆弱なキー作成またはキー管理
- 脆弱またはカスタムの暗号化
- チェックサムのなりすまし

### Magic xpa での対応

Magic xpa のプラットホームは自動的にすべてを暗号化し、メッセージの有効性と整合性を処理するため、このセクションで説明する 3 つの問題は Magic xpa の開発者では関係しません。

## 脆弱なキー作成またはキー管理

攻撃者が暗号化キーにアクセスしたり、暗号化キーを作成したりした場合、彼らは暗号化されたデータを解読することができます。キーが十分に管理されなかったり、ランダム形式で作成されていなかったりする場合、攻撃者はキーを入手する可能性があります。

### 脆弱なキー作成とキー管理の防止策：

- セキュアなキー管理を含むビルトインの暗号化ルーチンを使用してください。データ保護 API（DPAPI）は、Windows® 2000 以降の OS 上で提供される暗号化サービスの一例で、OS がキーを管理します。

- キーの作成または管理が必要な暗号化メカニズムを使用する場合、強力でランダムなキー作成機能を使用して、制限された場所（例えば、ACL で保護されたレジストリキー内において）でキーを保存してください。
- 更なる安全対策のために DPAPI を使用している暗号化キーを暗号化してください。
- 定期的にキーを期限切れにしてください。

## 脆弱またはカスタムの暗号化

暗号が解読された場合や、ブルートフォース攻撃に対して弱い場合、暗号化アルゴリズムはセキュリティ機能を提供できなくなります。特に十分にテストされない場合、カスタムアルゴリズムは脆弱になります。その代わりに、長年の厳しい攻撃や詳細な調査に耐え、公開されている有名な暗号化アルゴリズムを使用してください。

### 脆弱またはカスタマムの暗号化の防止策：

- 独自にカスタマイズしたアルゴリズムを作成しないでください。
- プラットホームによって提供される立証されたた暗号化サービスを使用してください。
- 解読されたアルゴリズムと解読技術についての情報を入手してください。

## チェックサムなりすまし

ネットワークで送られるメッセージに対応するデータ保全性を確保するために、ハッシュに依存しないでください。SHA1（Safe Hash Algorithm）や MD5(Message Digest algorithm) のようなハッシュは、横取りされ改ざんされる可能性があります。以下のベースを添付された MAC(Message Authentication Code) による Base64 のエンコーディングでの UTF-8 メッセージがあるとします。

```
Plaintext: Place 10 orders.
Hash: TOmUNdEQh13I09oTcaP4FYDX6pU=
```

攻撃者がネットワークをモニタすることによってメッセージを横取りすると、攻撃者はメッセージを改ざんすることができて、（使用したアルゴリズムを推測して）ハッシュを再計算することができます。たとえば、メッセージは以下のように変更することができます。

```
Plaintext: Place 100 orders.
Hash: oEDuJpv/ZtIU7BXDDNv17EAHeAU=
```

受取人がメッセージを処理し、ハッシュ化アルゴリズムによって平文（"Place 100 orders"）を解読してハッシュを再計算します。計算されたハッシュ値は、攻撃者によって計算されたものと同じになります。

この攻撃に対処するために、MAC または HMAC を使用してください。MACTripleDES（Message Authentication Code Triple Data Encryption Standard）アルゴリズムは MAC を計算し、HMACSHA1 は HMAC を計算します。両方とも、チェックサムを作成するために、キーを使用します。これらのアルゴリズムを使用することで、攻撃者は受信者として正しく計算されるチェックサムを作成するキーを知る必要があります。

# パラメータの改ざん

パラメータ改ざん攻撃は、クライアントとインターネット・アプリケーションの間で送られるパラメータ・データの変更に依存する攻撃です。これは、クエリ文字列やフォーム・フィールド、Cookie、HTTP ヘッダが含まれます。

パラメータ改ざんによる脅威には以下のものがあります。

- クエリ文字列操作
- フォーム・フィールド操作
- Cookie 操作
- HTTP ヘッダ操作

## クエリ文字列操作

HTTP GET によって渡されるクエリ文字列の値は、Web ブラウザの URL アドレスバーに表示される為、ユーザはクエリ文字列の値を簡単に操作することができます。アプリケーションがセキュリティを決定するためにクエリ文字列の値を使用している場合、または、値が金銭上の合計値のような機密に関わるデータを意味する場合、アプリケーションは攻撃に脆弱になります。

### クエリ文字列操作の防止策：

- サーバの機密上のロジックに影響するデータまたは機密データを含むクエリ文字列パラメータを使用することは避けてください。その代わりに、クライアントを特定するために、セッション ID を使用してください。また、サーバ上のセッションストア内に機密データを保存してください。
- フォームを送信する場合は、HTTP GET の代わりに、HTTP POST を使用してください。
- クエリ文字列パラメータを暗号化してください。

### Magic xpa での対応

Magic xpa RIA は Web ブラウザを使用しない為、RIA クライアント・モジュールで発行されるリクエスト・フォーマット上でエンドユーザがデータを取り出すことはできません。

## フォーム・フィールド操作

HTML フォーム・フィールドの値は、HTTP POST プロトコルを使用してサーバに平文で送られます。これは、表示と非表示のフォーム・フィールドが含まれる可能性があります。どのようなタイプのフォーム・フィールドでも簡単に変更することができます。そして、クライアント側の検証ルーチンを回避することができます。その結果、サーバ側のセキュリティ上の決定を行う為にフォーム・フィールドの入力値に依存するアプリケーションは、攻撃に弱くなります。

フォーム・フィールド操作の脅威に対処するためには、非表示のフォーム・フィールドを使用する代わりに、サーバ上の状態ストア内に維持される状態を参照する為に、セッション ID を使用してください。

### Magic xpa での対応

Web ブラウザも HTML も使用しないため、Magic xpa RIA では、アプリケーション・フローの外でアプリケーション・データやコントロールにアクセスして、操作することはできません。

## Cookie 操作

Cookie は、クライアントで変更される可能性があります。これは、固定およびメモリ常駐の Cookie に当てはまります。いくつかのツールは、攻撃者がメモリ常駐の Cookie の内容を修正するために利用できます。Cookie 操作は、通常、Web サイトへの不正アクセスを行う為に、Cookie を修正する攻撃です。

SSL によってネットワーク上の Cookie は保護されますが、クライアント PC での修正は防げません。Cookie 操作の脅威に対処するためには、Cookie を暗号化するか HMAC を使用してください。

### Magic xpa での対応

Magic xpa RIA のコンテキスト管理機能と Web ブラウザを使用しない特性により、コンテキスト管理の手段として Cookie の必要性がありません。

## HTTP ヘッダ操作

HTTP ヘッダは、クライアントとサーバの間で情報を受け渡しします。クライアントがリクエスト・ヘッダを作成し、サーバが応答ヘッダを作成します。アプリケーションが決定をするためにリクエスト・ヘッダに依存すると、アプリケーションは攻撃に弱くなります。

HTTP ヘッダにセキュリティ決定の基礎を置かないでください。たとえば、HTTP Referer は簡単に偽造されるので、クライアントがどこから来たかを決定するためにこの情報を信頼しないでください。

# 例外管理

クライアントに送信される例外によって内部実装の詳細が漏洩される場合があります。エンドユーザには意味がありませんが攻撃者にとって有用になります。例外処理を使用しなかったり、十分に実装されていないアプリケーションは、サービス拒否攻撃を受ける場合があります。

例外処理の脅威には以下のものがあります。

- 攻撃者による実装の詳細の漏洩
- サービス拒否

## 攻撃者による実装の詳細の漏洩

リッチな例外の詳細が攻撃者の手に落ちることを許すと、攻撃者が潜在的な脆弱性を利用して、将来の攻撃を計画する上での助けとなります。返される情報のタイプには、プラットホームのバージョンやサーバ名、SQL コマンド文字列、データベース接続文字列が含まれます。

### 攻撃者による実装の漏洩の防止策：

- アプリケーションのコード・ベース全体で例外処理を使用してください。
- アプリケーション境界に伝達できる例外を処理してロギングします。
- クライアントに一般的で、安全なエラーメッセージを返すようにしてください。

## サービス拒否

通常、故意に不正な形式の入力を渡すことによって、攻撃者はインターネット・アプリケーションを徹底的に調査します。これには、しばしば意図して 2 つの目的を持っています。一つは、例外を発生させ有益な情報を漏洩させることです。二つ目はインターネット・アプリケーションのプロセスをクラッシュさせることです。例外が適切にキャッチされて処理されないとこのようなことが発生する可能性があります。

### アプリケーション - レベルのサービス拒否の防止策：

- サーバでのすべての入力データを確実に検証してください。
- アプリケーションのコード・ベース全体で例外処理を使用してください。

# 監査とロギング

攻撃が実際に発生する前に、フットプリンティングまたはパスワードクラッキングのような企ての疑いを見つけるために監査やロギングを使用しなければなりません。これによって、否認の脅威も一緒に対応できます。ユーザによってトランザクションを実行したことを示す一連の同期されたログ・エントリが複数のサーバ上に存在すると、ユーザが処理を実行することを否認することが非常に難しくなります。

監査とロギングに関する脅威には以下のものがあります。

- ユーザが操作の実行を拒否する
- 攻撃者がトレースされることなくアプリケーションを利用する
- 攻撃者がトレースを隠す

## ユーザが操作の実行を否認する

否認の問題は、アクションを実行したか、トランザクションを開始したことを否定しているユーザに関係しています。すべてのユーザの活動を追跡し、記録することが確認できるようにするために、適所な防衛メカニズムを必要とします。

### 否認の防止策：

- Web サーバとデータベース・サーバ、そしてアプリケーション・サーバで監視とロギングを行ってください。
- キーイベント（例えばトランザクションやログイン／ログアウトのイベント）を記録してください。

- 実行元が確認できないため、共有アカウントを使用しないでください。

## 攻撃者が、トレースされることなくアプリケーションを利用する

システムとアプリケーションの各レベルの監査は、不審な活動が検出されずに実行されることを防止するために必要です。

### 不審な活動を検出するための防止策：

- 重要なアプリケーション・レベルの処理を記録してください。
- ログイン／ログアウトの各イベントやファイルシステムへのアクセス、オブジェクトへのアクセスの失敗を監視するため、プラットホーム・レベルの監視を使用してください。
- ログファイルをバックアップして、不審な活動の徴候に対して定期的に分析してください。

## 攻撃者がトレースを隠す

ログファイルは、攻撃者が自身の記録を隠すことがないようにする為に、適切に保護しなければなりません。

### 攻撃者による記録の隠蔽の防止策：

- 制限された ACL を使用してログファイルを保護してください。
- ログファイルの場所をデフォルトから変更してください。

### Magic xpa での対応

Magic xpa は、ビルトインされたロギング・メカニズムをアプリケーションサーバや MRB、リクエスタに提供します。このセクションで説明するように、アプリケーション・レベルで独自のログを作成することができます。

# 第 7 章　推薦事項

この章では、Magic xpa のモジュールに関係するセキュアな RIA アプリケーションの配備について推奨事項をまとめています。

## Magic xpa アプリケーションを保護する

### セキュアなレイヤ

セキュアな HTTP は、クライアントと Web サーバの間で使用する上で推奨されたプロトコルです。さらに、背後の通信のために Magic xpa の SSL サポートを利用することができます。SSL 経由で互いにデータをやり取りするために、アプリケーションサーバや MRB、リクエスタで環境設定を行うことができます。

### 暗号化されたデータ

確実に機密保持を行う為に、機密に関わるデータ暗号化を行う Magic xpa のビルトインサポート利用してください。

### ダイレクト SQL

Magic xpa のダイレクト SQL 機能の実装を選択した場合、ステートメントの整合性を妨害することができるユーザ入力によって、修正することができる SQL ステートメントを作成することは避けてください。

### エラー処理

Magic xpa のエラー処理メカニズムを使用して、アプリケーションの実行時のエラーを事前に対処してください。

### LDAP 機能

Magic xpa の認証機能（例えば LDAP や Microsoft の Active Directory®）との連携機能を利用してください。

### 権利メカニズム

確認された各ユーザに対して適切にアプリケーションの権利を与えるために、Magic xpa の権利メカニズムを使用してください。

### ベンダの署名

アプリケーション・ベンダの信頼性を証明して、アプリケーションのクライアント・モジュールのビルド処理の中で署名に利用する証明書を取得してください。