機器のサイバーセキュリティ確保のための
セキュリティ検証の手引き

別冊４　機器個別のセキュリティ検証プラクティス集

経済産業省　商務情報政策局

サイバーセキュリティ課

目次

1 背景と目的

1.1 背景

「機器のサイバーセキュリティ確保のためのセキュリティ検証の手引き」の本編では、セキュリティ対策の妥当性や脆弱性の有無を確認するために、機器のセキュリティ検証を行うことが重要であることを示した。例えば、製品の設計段階で脅威分析を行い、機器に起こりうる脅威やリスクを洗い出すことで、上流フェーズでセキュリティを考慮した作り込みを行うことができ、脆弱性の残存可能性の低減や、下流フェーズでの脆弱性検出による手戻りの工数削減が期待できる。また、出荷前の機器に対して検査及び診断を行うことで、脆弱性を出荷前に発見し、市場での悪用を回避することも期待できる。

セキュリティ検証を行う際には、セキュリティ検証サービスを提供する事業者に検証業務を依頼することが多い。しかしながら、検証に関する具体的なプラクティスや機器ごとの効果的な検証手法・留意事項等が事業者内で整理されていない場合があり、結果として、検証人材個人の暗黙知に依存したサービスが提供される場合もある。

1.2 本別冊の目的

令和４年度、経済産業省は、中小企業等が開発・販売する IoT 機器の対策の現状を把握するとともに、当該機器に対する検証の留意点や求められる対策項目を抽出する目的で、IoT 機器の脆弱性検証を希望する中小企業等を募集し、当該機器に対して検証を実施する実証事業を行った。この実証事業では、応募のあった製品のうち 74 社・155 製品に対して、検証サービス事業者による検証を行った。

本別冊は、この実証を通じて得られた結果を踏まえ、実証で応募のあった機器のうち代表的な機器ごとに、適用された検証手法やその結果検出された脆弱性の情報等の検証のプラクティスをまとめたものである。具体的には、以下の機器類型ごとに、検証プラクティスを整理している。

- UTM
- ゲートウェイ・ルータ
- ネットワークスイッチ
- モバイル端末
- スマートロック
- スマート家電
- ドローン
- ネットワークカメラ
- センサ・監視装置
- 産業用コントローラ

本別冊では、実証の結果を踏まえ、各機器に適用しうる検証手法をまとめるとともに、実証で実際に検出された深刻度の高い脆弱性の情報を示し、それぞれの脆弱性の検出に至ったプロセスを整理している。

さらに、当該脆弱性に対して想定される推奨事項を示すほか、各機器の検証に当たって留意すべき事項を示している。

1.3 本別冊の対象者・活用方法

本別冊では機器検証を実施する検証サービス事業者を主な対象としている。本別冊に記載の機器を対象に検証サービス事業者が検証を実施する際、適用する検証手法の検討や脆弱性の当たりをつけるために活用するほか、脆弱性が検出された際の推奨事項の検討に活用することが想定される。さらには、各機器の検証に当たっての留意事項を確認する際の活用も想定される。また、機器を開発するメーカにおいても本別冊を活用できる。具体的には、メーカの開発者が開発段階でセキュリティ対策を検討する際、どのような脆弱性に留意する必要があるかを確認するほか、検証担当者が社内で検証を実施する際の手引きとしての活用も期待される。

1.4 本別冊の構成

以降では、上述した各機器について、実証の結果を踏まえた検証のプラクティスを示す。各機器について、以下の項目を記載している。

1. **機器の概要・想定脅威**：当該機器の概要や想定されるユースケース、当該機器で考慮すべきセキュリティ脅威を記載。
2. **想定される検証環境**：当該機器の検証に当たって、構築すべき検証環境を記載。
3. **実証において適用された検証手法**：実証において、当該機器の検証に当たって適用された検証手法を記載。なお、詳細な検証手順については、本手引きの別冊 1「脅威分析及びセキュリティ検証の詳細解説書」[1]を参照する形式としている。
4. **実証において検出された代表的な脆弱性**：実証を通じて検出された脆弱性のうち、深刻度が高い脆弱性について、その概要や悪用された場合の影響について記載。
5. **想定される推奨事項**：検出された脆弱性に対して、どのような対策が推奨されるかを記載。
6. **検証に当たっての留意事項**：当該機器の検証に当たって、検証事業者が留意すべき事項を記載。

「付録２ 実証において検出された代表的な脆弱性」では、実証において検出された脆弱性のうち、深刻度が高い代表的な脆弱性について、脆弱性が検出された機器区分、当該脆弱性の概要と検証に当たって活用された情報（設計書、仕様書等の文書、ファームウェア、ソースコード、プロトタイプ、機器本体等）を示す。

検証の準備、依頼者とのコミュニケーション、検証結果報告書のとりまとめ等、すべての機器に共通して実施すべき事項については、本手引きの本編である「機器のサイバーセキュリティ確保のためのセキュリティ検証の手引き」を参照されたい。なお、「3. 実証において適用された検証手法」は、令和４年度の複数

[1] 経済産業省、機器のサイバーセキュリティ確保のためのセキュリティ検証の手引き 別冊 1 脅威分析及びセキュリティ検証の詳細解説書 https://www.meti.go.jp/press/2021/04/20210419003/20210419003-2.pdf

製品に対する実証において、当該機器類型に対して実際に適用されたすべての検証手法を記載しており、実際の検証において全ての検証手法の適用が必須で求められるものではない。検証事業者は、検証依頼者の予算、期間、要望等を踏まえて、適切な検証手法を選択・適用することが望まれる。また、メーカの検証担当者が本別冊を活用する際、各検証手法に求められるスキル・知識や工数が重要な指標となる。各検証手法に求められるスキル・知識のレベルや検証に要するコストのレベルについては、本手引きの別冊 3「検証人材の育成に向けた手引き」[2]の表 2-2 において整理しているため、必要に応じて、合わせて参照されたい。

[2] 経済産業省、機器のサイバーセキュリティ確保のためのセキュリティ検証の手引き 別冊 3 検証人材の育成に向けた手引き
https://www.meti.go.jp/press/2021/04/20210419003/20210419003-4.pdf

2 UTM に関するセキュリティ検証プラクティス

2.1 機器の概要・想定脅威

UTM（Unified Threat Management）とは、複数の異なるセキュリティ機能を一つのハードウェアに統合し、集中的にネットワーク管理を行う機器のことで、日本語では統合脅威管理と呼ばれる。ファイアウォール機能だけでなく、複数の脅威検知機能を有しており、ネットワークを包括的に防御することが可能となる。一般的な組織ネットワークの構成及び UTM の想定利用環境を図 2-1 に示す。

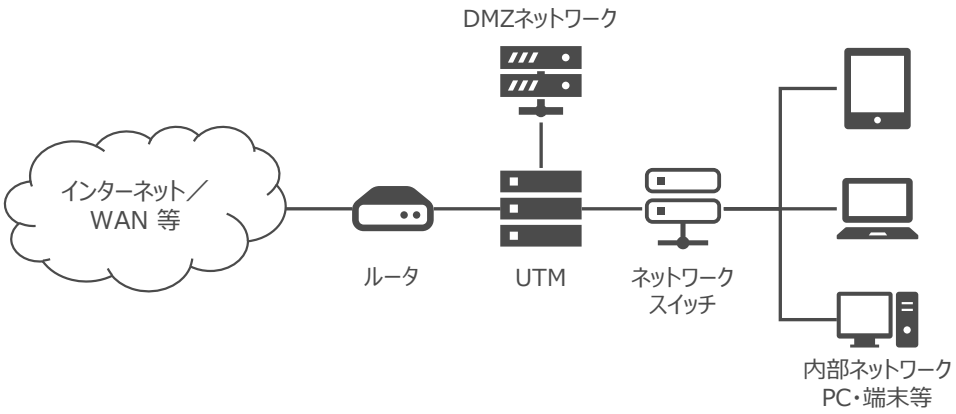

図 2-1 UTM の想定利用環境

UTM の運用においては、組織内外からの脅威を検知・防御し、組織におけるネットワーク機能を継続することが必要である。サイバー攻撃により UTM の機能を停止するためには、脆弱性を悪用して UTM 内部に侵入するほか、予期しないパケットやコマンドを UTM に送信して機能を停止させることが考えられる。侵入方法としては、開発用機能をバックドアとして悪用するほか、外部通信インタフェースの脆弱性を悪用する方法等が想定されるため、特に通信インタフェースに関する脅威に対する検証が求められる。

2.2 想定される検証環境

UTM の検証に当たって想定される検証環境は図 2-2 に示すとおりであり、検証事業者内に検証専用の LAN を構築して検証することが望まれる。

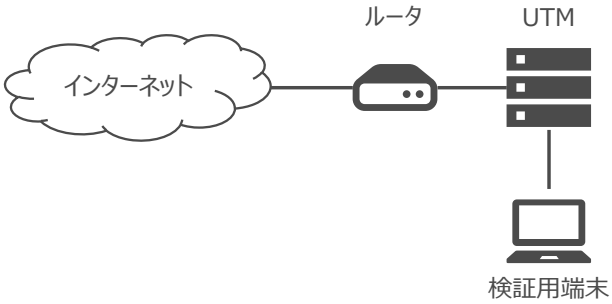

図 2-2 UTM に対する想定検証環境

2.3 実証において適用された検証手法

実証における UTM の検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。また、Web インタフェースに対して自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

検証対象機器の通信インタフェースや Web インタフェースに対して不正なデータを送信し、アプリケーシ

ョンや OS の応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。機能停止に対する防御性能を確認するためには、フラッド攻撃をはじめとする機器のリソースを消費される攻撃手法を試行し、サービス不能状態に陥ることがないかを確認することが有効である。また、検証対象機器が USB や SD カード等の外部記憶媒体に関するインタフェースを有している場合、不正なファイルを介したファイルファジングによって応答を確認することも望まれる。ファズデータの網羅性を担保することは困難であるため、検証範囲を絞るために、ファジング対象のインタフェースやファズデータのパラメータ数について、事前に検証依頼者と相談することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

検証対象機器の通信内容を取得し、平文の通信、暗号強度の低い通信、想定外の通信先への通信の有無を確認する。検証対象範囲について、最低限 Ethernet の通信に関するキャプチャを行うことが望まれる。また、ネットワークキャプチャの対象時間について、最低限 60 分程度の通信をキャプチャし、通信先・通信内容を確認することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（6） ファームウェア解析

検証対象機器が UART や SPI といったインタフェースを有している場合、当該インタフェースを介してファームウェアへのアクセスが可能となる場合があるため、ファームウェアの抽出可否を確認する。UART の場合、ブートローダコマンドの実行可否及びシェルアクセスの可否の確認が望まれる。SPI の場合、フラッシュメモリから SPI を経由してファームウェアが抽出可能であるかを確認することが望まれる。仮にファームウェアの抽出が可能であった場合に、当該ファームウェアファイルを解析し、ファイルシステムの取り出し可否を確認する。ファイルシステムの取り出しが可能であった場合に、ファイルの内容を解析し、機微な情報の取得が可能であるか否かを確認する。

検証の詳細手順：手引き別冊 1 第 4.3 節参照

2.4 実証において検出された代表的な脆弱性

実証事業で検証対象とした UTM において検出された代表的な脆弱性は以下のとおりである。

表 2-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	古いバージョンの OpenSSH における権限昇格の脆弱性	悪用されることで、機器に存在するプログラムが意図しない権限で実行される可能性がある。この結果、UTM の動作が停止し、当該 UTM を導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
2	Web 管理画面における OS コマンドインジェクションの脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、UTM の動作が停止し、当該 UTM を導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ OWASP ZAP のスパイダー機能を用い、検証対象機器の Web サービスにおいてアクセス可能な URL 一覧を作成する。 ・ ファジングの対象とする URL とパラメータのリストを作成する。パラメータリストの作成に当たっては、GitHub 等で公開されているペイロードリスト [3]を参考にする。 ・ 作成したパラメータリストに基づき、ファジングを実施する。 ・ ファズデータに対する Web サービスのレスポンスを踏まえ、脆弱性の有無を判断する。

[3] https://github.com/payloadbox, https://github.com/xmendez など

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
3	非暗号化通信によるファームウェアの更新	悪用されることで、ファームウェアファイルが窃取される可能性がある。また、中間者攻撃によってファームウェアが改ざんされ、この結果、UTM の動作が停止し、当該 UTM を導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Wireshark を用いたネットワークキャプチャを開始する。 ・ 検証対象機器の Web 管理画面からファームウェアの更新を行う。 ・ ファームウェア更新に係るパケットのキャプチャ結果を解析し、適切に暗号化されているかを確認する。

2.5 想定される推奨事項

表 2-1 で示した、実証事業で検証対象とした UTM において検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 2-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	古いバージョンの OpenSSH における権限昇格の脆弱性	・ 最新の OpenSSH にバージョンアップする。
2	Web 管理画面における OS コマンドインジェクションの脆弱性	・ シェルを起動できる言語機能の使用を避ける。 ・ 引数に対してチェックを行い、あらかじめ許可した処理のみ実行する。
3	非暗号化通信によるファームウェアの更新	・ SSL/TLS を用いて通信を暗号化する。

2.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、UTM 機能やアーキテクチャ、接続関係を調査・分析することが望まれる。具体的には、どのようにセッション維持管理がなされるか、どのように通信を監視しているか、通信内容に応じてどのような動作や通知がなされるか、どのように通信ファイルの展開処理が行われているか等の観点が含まれる。UTM が有する機能は多岐にわたるため、一つずつ検証を行うことは現実的ではない。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、UTM がどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。例えば、HTTP サーバに対する検証を行う場合は、HTTP におけるファズデータ（例：膨大な長さの URL）を生成する必要があり、HTTP サーバが処理しないレイヤのプロトコル（例：TCP/IP）におけるファズデータを生成しても有効な検証とはならない。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。

ファームウェア解析に当たって、ファームウェアの抽出可否の確認にはハードウェアを手動で解析する必要があり、費用が高くなる傾向がある。そのため、検証依頼者からファームウェアファイルを別途受領し、ファームウェアの抽出可否の確認はせず、ファームウェアファイルの解析のみを行う場合もある。

3 ゲートウェイ・ルータに関するセキュリティ検証プラクティス

3.1 機器の概要・想定脅威

ゲートウェイ・ルータは、WAN と LAN 等、異なるネットワークの環境に設置されることが多い機器であり、異なるネットワーク間や機器間の通信を中継する機器である。なお、ゲートウェイは、IoT 機器とクラウドサーバがデータの通信を行う際の中継機能を有する場合、IoT ゲートウェイと呼ばれる。一般的な組織ネットワークの構成及びゲートウェイ・ルータの想定利用環境を図 3-1 に示す。

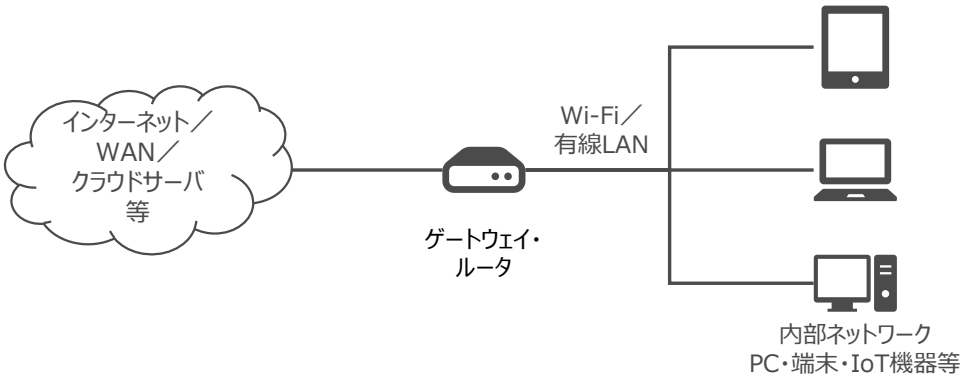

図 3-1 ゲートウェイ・ルータの想定利用環境

ゲートウェイ・ルータにおいて最も想定すべき脅威は、悪意ある第三者によってネットワーク機能や通信機能が阻害・停止されることにある。加えて、内部侵入されることで、機密情報（ID やパスワード、秘密鍵等）の窃取、ファームウェアの改ざん、正規ユーザ以外による不正利用といった脅威が考えられる。サイバー攻撃によりゲートウェイ・ルータの機能を停止するためには、脆弱性を悪用した内部侵入・権限昇格や、予期しないパケットやコマンドをゲートウェイ・ルータに送信することが考えられる。侵入方法としては、開発用機能をバックドアとして悪用するほか、外部通信インタフェースの脆弱性を悪用する方法等が想定されるため、特に通信インタフェースに関する脅威に対する検証が求められる。また、ゲートウェイ・ルータの機器設定用の Web コンソールが用意されている場合が多く、Web コンソールに内在しうる脆弱性として、クロスサイトスクリプティングや OS コマンドインジェクション等の入力検証の不備が考えられる。これらの脆弱性を洗い出す目的で、Web コンソールに対して、既知の攻撃手法に対する影響がないかを確認することが望ましい。

3.2 想定される検証環境

ゲートウェイ・ルータの検証に当たって想定される検証環境は図 3-2 に示すとおりであり、検証事業者内に検証専用の LAN を構築して検証することが望まれる。

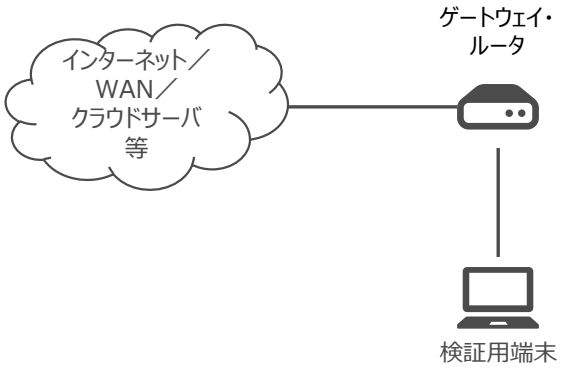

図 3-2 ゲートウェイ・ルータに対する想定検証環境

3.3 実証において適用された検証手法

実証におけるゲートウェイ・ルータの検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。また、Web インタフェースに対して自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能である

か、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

検証対象機器の通信インタフェースや Web インタフェースに対して不正なデータを送信し、アプリケーションや OS の応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。機能停止に対する防御性能を確認するためには、フラッド攻撃をはじめとする機器のリソースを消費される攻撃手法を試行し、サービス不能状態に陥ることがないかを確認することが有効である。また、検証対象機器が USB や SD カード等の外部記憶媒体に関するインタフェースを有している場合、不正なファイルを介したファイルファジングによって応答を確認することも望まれる。ファズデータの網羅性を担保することは困難であるため、検証範囲を絞るために、ファジング対象のインタフェースやファズデータのパラメータ数について、事前に検証依頼者と相談することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

検証対象機器の通信内容を取得し、平文の通信、暗号強度の低い通信、想定外の通信先への通信の有無を確認する。検証対象範囲について、最低限 Ethernet の通信に関するキャプチャを行うことが望まれる。また、ネットワークキャプチャの対象時間について、最低限 60 分程度の通信をキャプチャし、通信先・通信内容を確認することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（6） ファームウェア解析

検証対象機器が UART や SPI といったインタフェースを有している場合、当該インタフェースを介してファームウェアへのアクセスが可能となる場合があるため、ファームウェアの抽出可否を確認する。UART の場合、ブートローダコマンドの実行可否及びシェルアクセスの可否の確認が望まれる。SPI の場合、フラッシュメモリから SPI を経由してファームウェアが抽出可能であるかを確認することが望まれる。仮にファームウェアの抽出が可能であった場合に、当該ファームウェアファイルを解析し、ファイルシステムの取り出し可否を確

認する。ファイルシステムの取り出しが可能であった場合に、ファイルの内容を解析し、機微な情報の取得が可能であるか否かを確認する。また、OS、ファームウェア、ファイルシステムが SD カード等に格納されている場合、SD カード等に対する物理的アクセス可能性を確認するほか、SD カード等に格納されたデータが暗号化されているかを確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.3 節参照

（7） バイナリ解析

抽出したファームウェア等のバイナリファイルに対して、脆弱性の可能性のある箇所を確認する。脆弱性の可能性のある箇所が存在した場合、該当箇所で必ずしも脆弱性が発生するわけではない。そのため、リバースエンジニアリングツールを用いてバイナリファイルをデコンパイルし、ソースコードを確認した上で、当該脆弱性の悪用可能性を確認することが望ましい。なお、解析対象となるバイナリファイルについて、攻撃に当たってのヒントとなる情報を含む可能性があるデバッグ情報が残存する実行可能ファイルの優先度を高めて解析することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.4 節参照

3.4 実証において検出された代表的な脆弱性

実証事業で検証対象としたゲートウェイ・ルータにおいて検出された代表的な脆弱性は以下のとおりである。

表 3-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	古いバージョンの Squid における不正アクセスやリモートコード実行の脆弱性	悪用されることで、禁止されているリソースへのアクセスや任意のコードを実行される可能性がある。この結果、ゲートウェイ・ルータ内のファイルシステムの情報が漏えいするおそれがある。また、ゲートウェイ・ルータ内の任意のファイルの変更・追加・削除が行われ、ゲートウェイ・ルータが停止し、当該ゲートウェイ・ルータを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
2	古いバージョンの dnsmasq におけるバッファオーバーフローの脆弱性	悪用されることで、細工された不正なリクエストを受け、サービス拒否や任意のコードを実行される可能性がある。この結果、ゲートウェイ・ルータの動作が停止し、当該ゲートウェイ・ルータを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
3	古いバージョンの lighttpd における SQL インジェクションの脆弱性	悪用されることで、任意の SQL コマンドを実行される可能性がある。この結果、ゲートウェイ・ルータの動作が停止し、当該ゲートウェイ・ルータを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
4	バッファオーバーフローの脆弱性	悪用されることで、メモリ上の値が不正に書き換えられ、任意のコードが実行される可能性がある。この結果、ゲートウェイ・ルータの動作が停止し、当該ゲートウェイ・ルータを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ デバッグ情報が残存している実行可能ファイルが存在しないかを確認するため、ファームウェアに含まれるルートファイルシステムの全ファイルに対して file コマンドを実行し、デバッグ情報が残存している実行可能ファイルをリストアップする。 ・ リストアップした実行可能ファイルに対して、angr、cwe_checker 等を用いた静的解析を行い、脆弱性の可能性のある箇所を特定する。 ・ 特定した脆弱性の可能性のある箇所に対して、Ghidra、IDA Pro 等を用いて実行可能ファイルをデコンパイルし、ソースコードを確認し、脆弱性の悪用可能性を確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
5	古いバージョンの OS における不正アクセスやリモートコード実行の脆弱性	悪用されることで、禁止されているリソースへのアクセスや任意のコードを実行される可能性がある。この結果、ゲートウェイ・ルータ内のファイルシステムの情報が漏えいするおそれがある。また、ゲートウェイ・ルータ内の任意のファイルの変更・追加・削除が行われ、ゲートウェイ・ルータが停止し、当該ゲートウェイ・ルータを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ 検証対象機器で使用されている OS とバージョンを検証対象機器の仕様書等の文書から確認する。 ・ 確認した OS とバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
6	古いバージョンの Samba におけるリモートコード実行の脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、ゲートウェイ・ルータの動作が停止し、当該ゲートウェイ・ルータを使用している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
7	アップデートサーバ及びアップデートファイルの署名検証不備	悪用することで、改ざんしたアップデートファイルを配布し、悪意のある処理を含んだコードを実行される可能性がある。この結果、ゲートウェイ・ルータの動作が停止し、当該ゲートウェイ・ルータを使用している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ 正規サーバになりすまして、改ざんしたアップデートファイルを配布することが可能かを試行するため、検証用のサーバと任意の処理を含むコードを含むアップデートファイルを用意する。 ・ なりすましたサーバから検証用のアップデートファイルを検証対象機器に配布する。 ・ 検証用のアップデートファイルを配布したことに対する検証対象機器のレスポンスを踏まえ、脆弱性の有無を判断する。

3.5 想定される推奨事項

表 3-1 で示した、実証事業で検証対象としたゲートウェイ・ルータにおいて検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 3-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	古いバージョンの Squid における不正アクセスやリモートコード実行の脆弱性	・ 最新の Squid にバージョンアップする。
2	古いバージョンの dnsmasq におけるバッファオーバーフローの脆弱性	・ 最新の dnsmasq にバージョンアップする。
3	古いバージョンの lighttpd における SQL インジェクションの脆弱性	・ 最新の lighttpd にバージョンアップする。
4	バッファオーバーフローの脆弱性	・ バッファのサイズをチェックする処理をコードに追加する。
5	古いバージョンの OS における不正アクセスやリモートコード実行の脆弱性	・ 最新の OS にバージョンアップする。

項番	脆弱性の概要	推奨される対策事項
6	古いバージョンの Samba におけるリモートコード実行の脆弱性	・ 最新の Samba にバージョンアップする。
7	アップデートサーバ及びアップデートファイルの署名検証不備	・ アップデートファイルの取得において、サーバ証明書の検証を行う。 ・ アップデートファイルに対してデジタル署名を行い、アップデート時にアップデートファイルの署名検証を行う。

3.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、ゲートウェイ・ルータの機能やアーキテクチャを調査・分析することが望まれる。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、ゲートウェイ・ルータがどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。例えば、HTTP サーバに対する検証を行う場合は、HTTP におけるファズデータ（例：膨大な長さの URL）を生成する必要があり、HTTP サーバが処理しないレイヤのプロトコル（例：TCP/IP）におけるファズデータを生成しても有効な検証とはならない。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。

ファームウェア解析に当たって、ファームウェアの抽出可否の確認にはハードウェアを手動で解析する必要があり、費用が高くなる傾向がある。そのため、検証依頼者からファームウェアファイルを別途受領し、ファームウェアの抽出可否の確認はせず、ファームウェアファイルの解析のみを行う場合もある。なお、検証を実施するために、検証対象機器を分解することがある。検証対象機器を分解した後に、原状復帰することが困難な可能性がある場合は、事前に検証依頼者へ伝え、検証依頼者からの合意を得た後に検証を実施することが望まれる。

バイナリ解析に当たって、抽出・受領したファームウェア等のバイナリファイルについて、脆弱性の可能性のある箇所を特定するだけでなく、特定した箇所の脆弱性の再現性（悪用可能性）を確認することが望まれる。

また、IoT ゲートウェイのような機器がクラウドサーバと接続しているケースが考えられる。本手引きでは、クラウドサーバを対象とする検証は対象外としているが、クラウドサーバにおいては、悪意ある第三者によっ

てデータの改ざんや搾取、機能を停止される等の脅威も想定される。もし、検証依頼者の依頼を踏まえクラウドサーバに対する検証を行う場合、通信インタフェースやクラウドサーバの管理コンソールに対する検証を実施することが望ましい。なお、稼働中のクラウドサーバに対して検証を行う場合、他のサービスに影響を与えないか等を事前に検証依頼者と協議し、双方の合意の下で検証を実施することが必要である。

4 ネットワークスイッチに関するセキュリティ検証プラクティス

4.1 機器の概要・想定脅威

ネットワークスイッチとは、受信したデータの宛先を確認し、ネットワークスイッチに接続された機器へのデータの転送可否を判断し、転送する機器である。受信したデータを接続されたすべての機器へ転送する機器と比べて、通信経路上に余計なデータが流れることを防ぎ、ネットワークの性能を向上させることが可能となる。一般的な組織ネットワークの構成及びネットワークスイッチの想定利用環境を図 4-1 に示す。

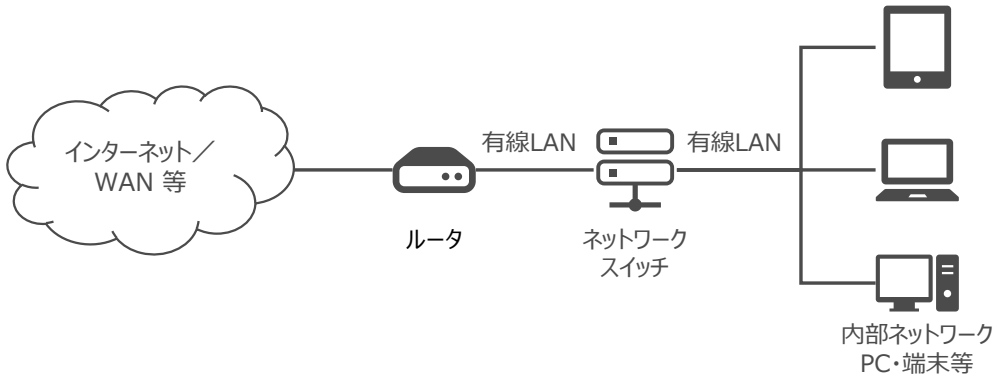

図 4-1 ネットワークスイッチの想定利用環境

ネットワークスイッチにおいて最も想定すべき脅威は、悪意ある第三者によってネットワーク機能や通信機能が阻害・停止されることにある。加えて、内部侵入されることで、機密情報（ID やパスワード等）の窃取、ファームウェアの改ざん、正規ユーザ以外による不正利用といった脅威が考えられる。サイバー攻撃によりネットワークスイッチの機能を停止するためには、脆弱性を悪用した内部侵入・権限昇格や、予期しないパケットやコマンドをネットワークスイッチに送信することが考えられる。侵入方法としては、開発用機能をバックドアとして悪用するほか、外部通信インタフェースの脆弱性を悪用する方法等が想定されるため、特に通信インタフェースに関する脅威に対する検証が求められる。また、ネットワークスイッチの機器設定用の Web コンソールが用意されている場合が多く、Web コンソールに内在しうる脆弱性として、クロスサイトスクリプティングや OS コマンドインジェクション等の入力検証の不備が考えられる。これらの脆弱性を洗い出す目的で、Web コンソールに対して、既知の攻撃手法に対する影響がないかを確認することが望ましい。

4.2 想定される検証環境

ネットワークスイッチの検証に当たって想定される検証環境は図 4-2 に示すとおりであり、検証事業者内に検証専用の LAN を構築して検証することが望まれる。

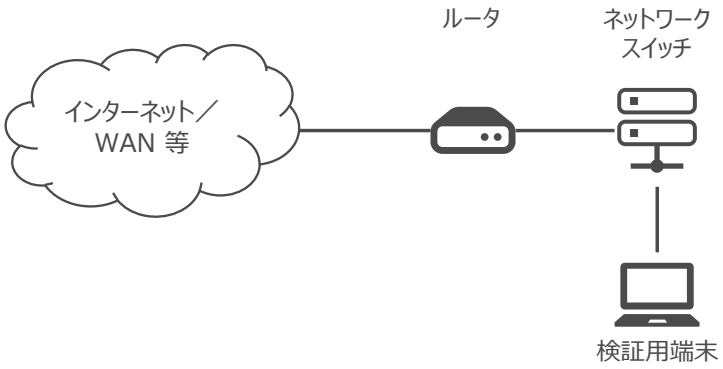

図 4-2 ネットワークスイッチに対する想定検証環境

4.3 実証において適用された検証手法

実証におけるネットワークスイッチの検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。また、Web インタフェースに対して自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能である

か、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

検証対象機器の通信インタフェースや Web インタフェースに対して不正なデータを送信し、アプリケーションや OS の応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。機能停止に対する防御性能を確認するためには、フラッド攻撃をはじめとする機器のリソースを消費される攻撃手法を試行し、サービス不能状態に陥ることがないかを確認することが有効である。また、Web インタフェースに対しては、開発者が意図しない使用方法（意図しない拡張子のファイルをアップロードするなど）を試行し、レスポンスを確認することで、脆弱性の有無を確認することが有効である。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

検証対象機器の通信内容を取得し、平文の通信、暗号強度の低い通信、想定外の通信先への通信の有無を確認する。検証対象範囲について、最低限 Ethernet の通信に関するキャプチャを行うことが望まれる。また、ネットワークキャプチャの対象時間について、最低限 60 分程度の通信をキャプチャし、通信先・通信内容を確認することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

4.4 実証において検出された代表的な脆弱性

実証事業で検証対象としたネットワークスイッチにおいて検出された代表的な脆弱性は以下のとおりである。

表 4-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	古いバージョンの Dropbear における情報の漏えいや改ざんの脆弱性	悪用されることで、情報漏えいや情報の改ざんされる可能性がある。この結果、ネットワークスイッチの動作が停止し、当該ネットワークスイッチを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
2	Web 管理画面に対する非暗号化状態での通信	ネットワーク上を流れる通信を盗聴されることで、Web 管理画面へのログイン ID やパスワード、機器操作に関するコマンド実行結果の通信内容といった機密情報が漏えいする可能性がある。これにより、攻撃者は、ネットワークスイッチへの不正アクセスや攻撃手口のヒントの取得が可能となる。この結果、サイバー攻撃を受けることで、ネットワークスイッチの動作が停止し、当該ネットワークスイッチを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Wireshark を用いたネットワークキャプチャを開始する。 ・ 検証対象機器の Web 管理画面へのログインや機器操作に関するコマンド実行の操作を行う。 ・ ログインや機器操作に関するコマンド実行の通信に係るパケットのキャプチャ結果を解析し、適切に暗号化されているかを確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
3	Web 管理画面におけるクロスサイトスクリプティングの脆弱性	悪用されることで、正規ユーザは改ざんされた Web ページへ誘導され、任意のコードが実行される可能性がある。この結果、マルウェアに感染し、ネットワークスイッチの動作が停止し、当該ネットワークスイッチを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ OWASP ZAP のスパイダー機能を用い、検証対象機器の Web サービスにおいてアクセス可能な URL 一覧を作成する。 ・ ファジングの対象とする URL とパラメータのリストを作成する。パラメータリストの作成に当たっては、GitHub 等で公開されているペイロードリストを参考にする。 ・ 作成したパラメータリストに基づき、ファジングを実施する。 ・ ファズデータに対する Web サービスのレスポンスを踏まえ、脆弱性の有無を判断する。
4	Web 管理画面における画像アップロード機能において任意のファイルをアップロード可能	悪用されることで、Web 管理画面上で悪質なプログラムファイルが設置される可能性がある。この結果、マルウェアに感染し、ネットワークスイッチの動作が停止し、当該ネットワークスイッチを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Web 管理画面における画像アップロード機能において、開発者が意図しない拡張子のファイルアップロードが可能かを試行するため、検証用のファイルを作成する。 ・ 作成した検証用ファイルを画像アップロード機能においてアップロードする。 ・ 検証用ファイルをアップロードしたことに対する Web 管理画面のレスポンスを踏まえ、脆弱性の有無を判断する。

4.5 想定される推奨事項

表 4-1 で示した、実証事業で検証対象としたネットワークスイッチにおいて検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 4-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	古いバージョンの Dropbear における情報の漏えいや改ざんの脆弱性	・ セキュリティ修正プログラムを適用する。 ・ 最新の Dropbear にバージョンアップする。
2	Web 管理画面に対する非暗号化状態での通信	・ SSL/TLS を用いて通信を暗号化する。
3	Web 管理画面におけるクロスサイトスクリプティングの脆弱性	・ Web アプリケーションが動的に出力するすべてのパラメータに対して、Web ページの表示や動作に影響する特別な意味をもつ記号や文字列[4]にエスケープ処理を行う。なお、開発言語が実装している標準関数や標準ライブラリ・クラスが、エスケープ処理の機能を提供している場合は、これらの機能を利用し対策することも可能である。
4	Web 管理画面における画像アップロード機能において任意のファイルをアップロード可能	・ アップロード可能なファイルを開発者が意図する拡張子のみへ限定し、規定外の拡張子のファイルのアップロードを禁止する。 ・ アップロードされるファイルに対するウイルススキャン処理を行う。

4.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、ネットワークスイッチの機能やアーキテクチャを調査・分析することが望まれる。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、ネットワークスイッチがどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。例えば、HTTP サーバに対する検証を行う場合は、HTTP におけるファズデータ（例：膨大な長さの URL）を生成する必要があり、HTTP サーバが処理しないレイヤのプロトコル（例：TCP/IP）

[4] 「<」、「>」、「&」、「”」、「’」、「JavaScript:」、「;」、「（」、「）」、「//」、「¥」、「¥0」、「%00」 など

におけるファズデータを生成しても有効な検証とはならない。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。

5 モバイル端末に関するセキュリティ検証プラクティス

5.1 機器の概要・想定脅威

モバイル端末とは、充電式のバッテリーを内蔵し、屋外など電源のない場所でも電池が無くなるまで使用することができ、容易に持ち運ぶことができる携帯型の情報機器端末であり、スマートフォン、タブレット端末、ノートPC等を含む。一般的なネットワークの構成及びモバイル端末の想定利用環境を図 5-1 に示す。

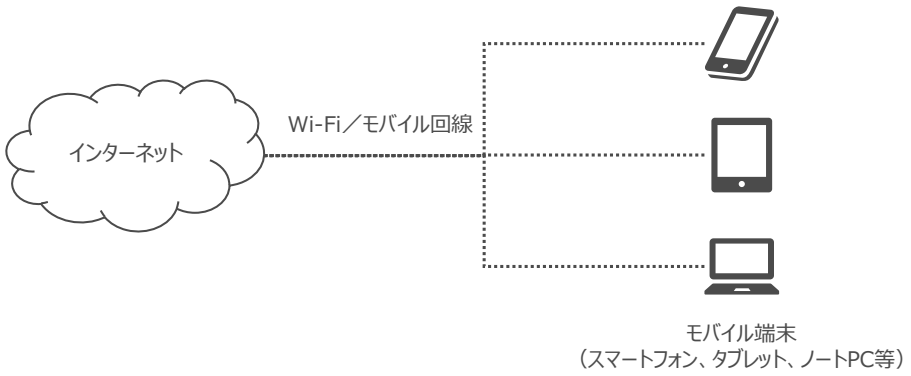

図 5-1 モバイル端末の想定利用環境

モバイル端末において最も想定すべき脅威は、第三者による端末の不正操作や利用者情報の第三者による窃取である。これらの脅威につながりうる脆弱性の有無を確認することが必要となる。例えば、端末の設定情報に関する調査では、不正なアプリケーションの起動が許可されている設定となっていないかを確認することが望まれる。また、攻撃者が不正なデバイスドライバに置き換えて利用者のデータをフックし、データが窃取される攻撃も想定されるため、Secure Boot や署名検証の有無についても確認することが望まれる。その他の脅威としては、初期インストールされたアプリケーションや、アップデート用のソフトウェアに対して、プログラムの脆弱性を悪用して管理者権限を奪取し、不正操作や利用者の情報を窃取が想定される。ソースコードを入手できる場合、ソースコード解析によって入力検証の不備や不適切なデータ処理等の代表的な脆弱性の有無を確認することが望まれる。ソースコードが入手できない場合、ブラックボックスでの検証となるため、優先度の高い脅威に対する検証に限定して実施する必要があり、権限設定の妥当性、不正処理の実行可能性、外部からの悪用可能性等を重点的に確認することが望まれる。また、初期インストールされたアプリケーションに対する検証だけではなく、不正なアプリケーションがインストールできるかという観点も、情報窃取の脆弱性を調べる上で重要な観点である。

5.2 想定される検証環境

モバイル端末の検証に当たって想定される検証環境は図 5-2 に示すとおりであり、検証用端末を検証対象機器に接続して検証することが望まれる。

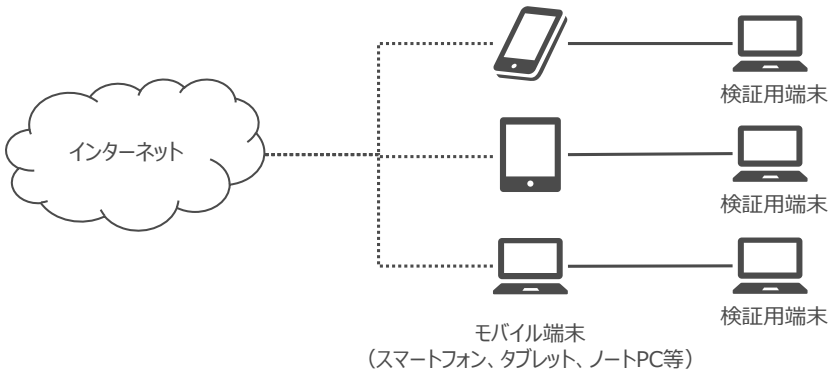

図 5-2 モバイル端末に対する想定検証環境

5.3 実証において適用された検証手法

実証におけるモバイル端末の検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。また、初期インストールされたアプリケーションに対して自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再

現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

検証対象機器の通信インタフェースや初期インストールされたアプリケーションに対して不正なデータを送信し、アプリケーションや OS の応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。機能停止に対する防御性能を確認するためには、フラッド攻撃をはじめとする機器のリソースを消費される攻撃手法を試行し、サービス不能状態に陥ることがないかを確認することが有効である。また、検証対象機器が USB や SD カード等の外部記憶媒体に関するインタフェースを有している場合、不正なファイルを介したファイルファジングによって応答を確認することも望まれる。ファズデータの網羅性を担保することは困難であるため、検証範囲を絞るために、ファジング対象のインタフェースやファズデータのパラメータ数について、事前に検証依頼者と相談することが望まれる。加えて、スマートフォンやタブレット端末のようなタッチパネル機能を持つ場合は、正規ユーザが通常行わないようなタッチパネル操作によって、機密情報が閲覧できないか等の脆弱性の有無を確認することも望まれる。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

検証対象機器の USB や無線 LAN 等のインタフェースから通信内容を取得し、平文の通信、暗号強度の低い通信、想定外の通信先への通信の有無を確認する。ネットワークキャプチャの対象時間について、最低限 60 分程度の通信をキャプチャし、通信先・通信内容を確認することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（6） ファームウェア解析

検証対象機器のインタフェースを介してファームウェアへのアクセスが可能となる場合があるため、ファームウェアの抽出可否を確認する。仮にファームウェアの抽出が可能であった場合に、当該ファームウェアファイルを解析し、ファイルシステムの取り出し可否を確認する。ファイルシステムの取り出しが可能であった場合に、ファイルの内容を解析し、機微な情報の取得が可能であるか否かを確認する。また、OS、ファームウェア、ファイルシステムが SD カード等に格納されている場合、SD カード等に対する物理的なアクセスが可能

であるかを確認するほか、SD カード等に格納されたデータが暗号化されているかを確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.3 節参照

（7） バイナリ解析

抽出したファームウェア等のバイナリファイルに対して、脆弱性の可能性のある箇所を確認する。脆弱性の可能性のある箇所が存在した場合、該当箇所で必ずしも脆弱性が発生するわけではない。そのため、リバースエンジニアリングツールを用いてバイナリファイルをデコンパイルし、ソースコードを確認し、脆弱性の悪用可能性を確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.4 節参照

5.4 実証において検出された代表的な脆弱性

実証事業で検証対象としたモバイル端末において検出された代表的な脆弱性は以下のとおりである。

表 5-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	初期インストールされた Web アプリケーションにおけるディレクトリトラバーサルの脆弱性	悪用されることで、Web アプリケーション内の任意のファイルにアクセスされる可能性がある。この結果、任意のファイルに格納された重要情報が漏えいするおそれがある。	・ Web アプリケーションにアクセスし、ファイルパスに任意のパスを入力する。 ・ 入力したファイルパスに格納されたファイルの内容が表示されるかを確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
2	タッチパネル操作によって機密情報の閲覧が可能	悪用されることで、ファイルシステムに格納されたパスワードファイル等の機密情報を閲覧される可能性がある。この結果、閲覧されたパスワード等の機密情報を利用し、さらなる機密情報が漏えいするおそれがある。	・ モバイル端末で正規ユーザが利用しないようなアイコンやメニューが存在しないかを確認する。 ・ 確認したアイコンやメニューから機密情報を閲覧できないかを確認する。
3	誰もが匿名で利用可能なサーバが稼働	悪用されることで、ディレクトリやファイルの操作（閲覧、作成、削除等）が行われる可能性がある。この結果、任意のファイルのダウンロードによる情報漏えい、ファイルの改ざん・消去されるおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
4	初期インストールされた Web アプリケーションにおいて権限を改ざんした操作が可能	悪用されることで、一般ユーザの権限を越えた操作（ユーザの作成・削除等）が行われる可能性がある。この結果、当該モバイル端末を導入している企業の業務が停止するおそれがある。	・ Web アプリケーションに一般ユーザでログインし、ログイン時に発行されるアクセストークンを Burp Suite 等のローカルプロキシツールで取得する。 ・ 取得したアクセストークンのペイロード部分に含まれる権限設定の内容を改ざんする。 ・ JWT（JSON Web Token）等のツールを利用し、改ざんしたペイロードの内容でアクセストークンを生成する。 ・ Web アプリケーションのユーザ一覧を表示させ、ユーザ検索ボタンを押下する。検索ボタンを押下した際のリクエストに対して、ローカルプロキシツールを使用し、ユーザを削除するリクエストの内容へ書き換えて送信する。なお、アクセストークンはした改ざんしたものを使用する。 ・ ユーザが削除されたことを確認する。（一般ユーザの権限を越えた操作が可能であることを確認する）

5.5 想定される推奨事項

表 5-1 で示した、実証事業で検証対象としたモバイル端末において検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 5-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	初期インストールされた Web アプリケーションにおけるディレクトリトラバーサルの脆弱性	・ 外部から入力される値をファイル名として、ファイルを読み込まないようにする。ファイルを開く際は、固定のディレクトリを指定し、ファイル名にディレクトリ名が含まれないようにする。 ・ Web サーバ内のファイルへのアクセス権限設定を正しく管理し、Web アプリケーションから必要以上のファイルにアクセスできないようにする。

項番	脆弱性の概要	推奨される対策事項
2	タッチパネル操作によって機密情報の閲覧が可能	・ 正規ユーザに対して、タッチパネルの操作を必要最低限に制限する。
3	誰もが匿名で利用可能なサーバが稼働	・ 脆弱性のあるサーバの機能を停止し、特定ユーザのみ該当サーバの利用が可能なように設定する。
4	初期インストールされた Web アプリケーションにおいて権限を改ざんした操作が可能	・ アクセストークンの署名に用いる重要情報はサーバからのレスポンスデータに含めず、ユーザに公開されないようにする。

5.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、モバイル端末の機能（初期インストールされたアプリケーションの有無を含む）やアーキテクチャを調査・分析することが望まれる。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、モバイル端末がどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。

ファームウェア解析に当たって、ファームウェアの抽出可否の確認にはハードウェアを手動で解析する必要があり、費用が高くなる傾向がある。そのため、検証依頼者からファームウェアファイルを別途受領し、ファームウェアの抽出可否の確認はせず、ファームウェアファイルの解析のみを行う場合もある。なお、検証を実施するために、検証対象機器を分解することがある。検証対象機器を分解した後に、原状復帰することが困難な可能性がある場合は、事前に検証依頼者へ伝え、検証依頼者からの合意を得た後に検証を実施することが望まれる。

バイナリ解析に当たって、抽出・受領したファームウェア等のバイナリファイルについて、脆弱性の可能性のある箇所を特定するだけでなく、特定した箇所の脆弱性の再現性（悪用可能性）を確認することが望まれる。

6 スマートロックに関するセキュリティ検証プラクティス

6.1 機器の概要・想定脅威

スマートロックとは、電気通信可能な錠で、スマートフォン等を用いて錠の開閉を行う機器である。市販の多くのスマートロックは Bluetooth の施錠・解錠機能を有しており、専用アプリを導入したスマートフォン等によって施解錠することができる。そのほか、Wi-Fi 接続機能を有したスマートロックの場合、同様に専用アプリを介して遠隔から施錠・解錠可能な場合もある。また、入退室管理の機能を有するスマートロックの場合、ユーザごとの入退室状況を管理するための Web アプリケーションも含めて構成されることがある。一例として、ネットワークの構成及びスマートロックの想定利用環境を図 6-1 に示す。

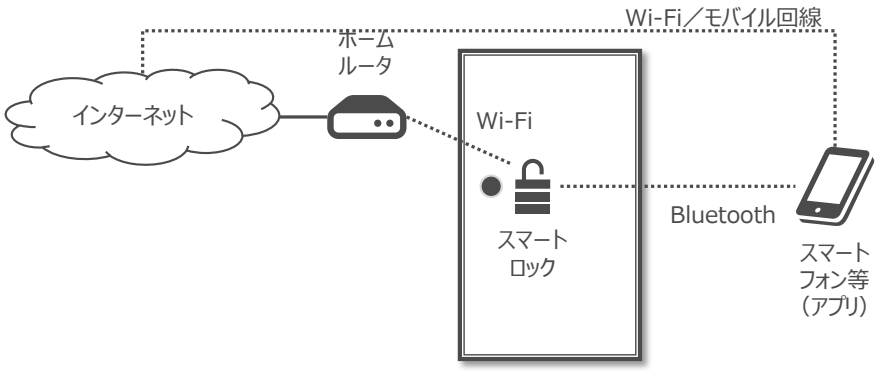

図 6-1 スマートロックの想定利用環境

スマートロック特有の想定脅威として、悪意ある第三者によってスマートロックが施解錠されることが挙げられる。この脅威につながりうる脆弱性の有無を確認するためには、スマートフォン等の端末を用いて施錠・解錠するという特徴を踏まえると、検証すべき観点は、不正なスマートフォンアプリによって施錠・解錠されないか、既存のスマートフォンアプリの不正操作によって解錠されないか、スマートフォン・スマートロック間の通信の盗聴や中間者攻撃によって解錠されないか等が考えられる。また、スマートロックは小型なため、十分なメモリ容量を有していないことも想定されるため、オーバーフロー攻撃によって、スマートロックの機能が停止され、本来の動作である施錠・解錠を受け付けない可能性もある。このような脆弱性についても併せて検証することが望まれる。

6.2 想定される検証環境

スマートロックの検証に当たって想定される検証環境は図 6-2 に示すとおりであり、検証事業者内に検証専用の LAN を構築して検証することが望まれる。なお、スマートロックにおける脅威は、スマートロック本体のみではないため、施錠・解錠のためのスマートフォンアプリや入退室管理のための Web アプリケーション等も検証対象とすることが望まれる。

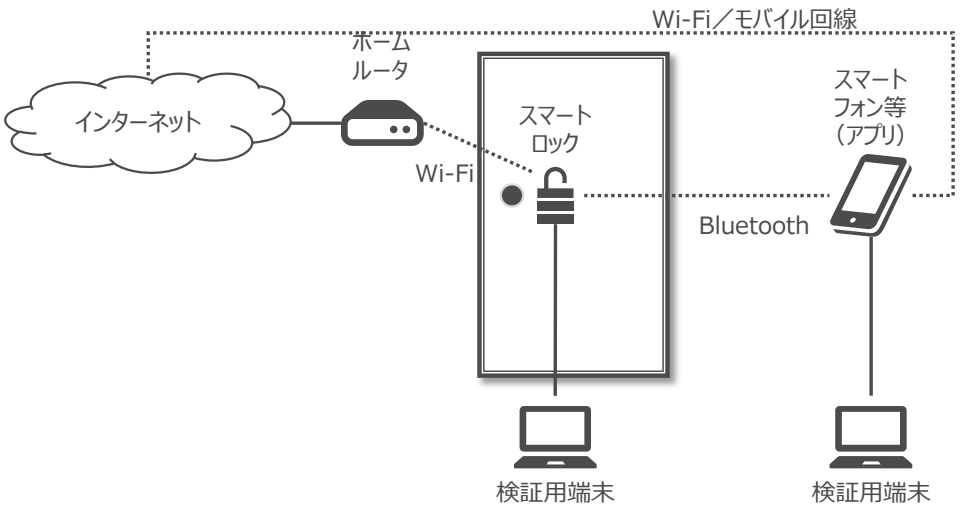

図 6-2 スマートロックに対する想定検証環境

6.3 実証において適用された検証手法

実証におけるスマートロックの検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。また、施錠・解錠のためのスマートフォンアプリに対しては、静的解析・動的解析のツールを使用することが

効果的である。静的解析・動的解析においては、スマートロックの施錠・解錠が第三者によって可能かを確認することが必要となるため、特に通信に関するプログラムを解析することが望まれる。攻撃者の視点に立てば、悪用できるアプリに制限はない。そのため、施錠・解錠のためのスマートフォンアプリが、Android や iOS 等の複数の OS で用意されている場合には、複数 OS のアプリに対して解析を行うことが望まれる。Web アプリケーションに対しては、自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。なお、自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

スマートロックや施錠・解錠のためのスマートフォンアプリの通信インタフェースや Web アプリケーションに対して不正なデータを送信し、OS や Web アプリケーションの応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。また、スマートロックが十分なメモリ容量を有していない場合、オーバーフロー攻撃によって、スマートロックの機能が停止される可能性があるかを確認する。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

スマートロックと施錠・解錠のためのスマートフォンアプリ間の通信の盗聴や中間者攻撃の可能性を確認する。スマートロックの施錠・解錠のための通信は Bluetooth で行われることが多く、この通信を盗聴できるかをパケットキャプチャにより検証することが望まれる。施錠・解錠のための通信をキャプチャする方法として、スマートロックになりすまし、なりすました機器を正規のスマートフォン等と接続することで通信内容を取得することが考えられる。なりすましの結果、スマートロックの施錠・解錠に必要なリクエストを入手できる場合、そのリクエストを悪用し、スマートロックに送信することで、施錠・解錠が可能かを検証することができる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（6）　ファームウェア解析

検証対象機器のインタフェースを介してファームウェアへのアクセスが可能となる場合があるため、ファームウェアの抽出可否を確認する。仮にファームウェアの抽出が可能であった場合に、当該ファームウェアファイルを解析し、ファイルシステムの取り出し可否を確認する。ファイルシステムの取り出しが可能であった場合に、ファイルの内容を解析し、機微な情報の取得が可能であるか否かを確認する。また、OS、ファームウェア、ファイルシステムが SD カード等に格納されている場合、SD カード等に対する物理的なアクセスが可能であるかを確認するほか、SD カード等に格納されたデータが暗号化されているかを確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.3 節参照

（7）　ソースコード解析

施錠・解錠のためのスマートフォンアプリのソースコードを確認し、脆弱性が含まれていないかを静的に解析する。ソースコード解析では、自動化ツールを活用してソースコードに含まれる特定のパターンを抽出することで、脆弱性を検出するほか、また、自動化ツールを使用せず、検証作業者によるマニュアル診断も想定される。ソースコードの確認観点は、機微情報がハードコートされていないか、ビルド時にソースコードが難読化されているか等が挙げられる。なお、ビルド時にソースコードが難読化されていない場合、攻撃者がプログラムを解析する目的でデコンパイルした場合、ソースコードを解析しやすく、この場合、攻撃するためにヒントとなる情報の窃取や検証対象機器への想定していない操作による攻撃が実施される可能性がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。

6.4 実証において検出された代表的な脆弱性

実証事業で検証対象としたスマートロックにおいて検出された代表的な脆弱性は以下のとおりである。

表 6-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	Web アプリケーションにおいて他ユーザの情報変更が可能	悪用されることで、ユーザのメールアドレスやパスワード等を改ざんされる可能性がある。この結果、メールアドレスやパスワード等を変更されたユーザは Web アプリケーションへログインできなくなり、スマートロックによる入退室管理の機能が停止するおそれがある。	・ Web アプリケーションへログインし、ユーザの情報変更画面へアクセスする。 ・ ユーザの情報変更画面の URL に含まれるユーザ ID を他ユーザの ID に改ざんして送信し、他ユーザの情報変更画面へアクセスする。 ・ アクセスした他ユーザの情報変更画面にて、メールアドレス、パスワード等を変更できるか確認する。
2	Web アプリケーションにおけるクロスサイトスクリプティングの脆弱性	悪用されることで、正規ユーザは改ざんされた Web ページへ誘導され、任意のコードが実行される可能性がある。この結果、マルウェアに感染し、スマートロックによる入退室管理の機能が停止し、当該スマートロックを導入しているユーザが入退室できなくなるおそれがある。	・ OWASP ZAP のスパイダー機能を用い、検証対象機器の Web アプリケーションにおいてアクセス可能な URL 一覧を作成する。 ・ ファジングの対象とする URL とパラメータのリストを作成する。パラメータリストの作成に当たっては、GitHub 等で公開されているペイロードリスト [5]を参考にする。 ・ 作成したパラメータリストに基づき、ファジングを実施する。 ・ ファズデータに対する Web サービスのレスポンスを踏まえ、脆弱性の有無を判断する。

[5] https://github.com/payloadbox, https://github.com/xmendez など

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
3	古いバージョンの systemd におけるリモートコード実行の脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、スマートロックの動作が停止し、当該スマートロックを導入しているユーザが入退室できなくなるおそれがある。	・ Nmap、シリアルコンソールを用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
4	Web アプリケーションにおけるユーザのパスワード攻撃の脆弱性	文字数が短いパスワードは、パスワードクラッキングツールの実行等により、パスワードの解析が容易に行える可能性がある。パスワードを解析された結果、攻撃者が正規ユーザになりすましてログインを行い、正規ユーザの権限で利用可能な様々な機能を行われるおそれがある。	・ Web アプリケーションへログインし、パスワード変更画面へアクセスする。 ・ パスワードを 1 文字等の短いパスワードへ変更することが可能かを確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
5	MQTT（Message Queuing Telemetry Transport）通信における認証制御の不備	悪用されることで、当該スマートロックの操作に関わる情報を取得される可能性がある。この結果、リモートで当該スマートロックを操作されるおそれがある。	・ 検証対象機器と MQTT ブローカー（サーバ）間の通信をネットワークキャプチャし、MQTT の認証情報を取得する。 ・ MQTT の認証情報を使用し、Mosquitto 等のツール上で、検証対象機器の状態取得や制御に使用するトピック名やメッセージ内容を表示する。

6.5 想定される推奨事項

表 6-1 で示した、実証事業で検証対象としたスマートロックにおいて検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 6-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	Web アプリケーションにおいて他ユーザの情報変更が可能	・ 正規ユーザに対して、アクセス可能な画面や実行可能な処理を必要最低限にする。
2	Web アプリケーションにおけるクロスサイトスクリプティングの脆弱性	・ Web アプリケーションが動的に出力するすべてのパラメータに対して、Web ページの表示や動作に影響する特別な意味をもつ記号や文字列[6]にエスケープ処理を行う。なお、開発言語が実装している標準関数や標準ライブラリ・クラスが、エスケープ処理の機能を提供している場合は、これらの機能を利用し対策することも可能である。
3	古いバージョンの systemd におけるリモートコード実行の脆弱性	・ 最新の systemd にバージョンアップする。

[6] 「<」、「>」、「&」、「”」、「’」、「JavaScript:」、「;」、「（」、「）」、「//」、「¥」、「¥0」、「%00」 など

項番	脆弱性の概要	推奨される対策事項
4	Web アプリケーションにおけるユーザのパスワード攻撃の脆弱性	・ パスワードとして登録可能な文字数の下限を 12 文字以上と制限する。なお、パスワードの要件に関しては、国内外の様々な団体で見解[7]があり、統一的なルールはない。そのため、本対策事項は一例であることに注意する必要がある。
5	MQTT 通信における認証制御の不備	・ MQTT 通信に必要な認証情報を固定しないよう、スマートロックごとに異なるアカウントを発行する。また、発行したアカウントに対して必要最低限の権限のみを設定する。

6.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、スマートロックの機能（入退室管理機能を有するかどうかを含む）やアーキテクチャを調査・分析することが望まれる。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、スマートロックがどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。施錠・解錠のためのスマートフォンアプリに関しては、Android と iOS 等の複数の OS に対応するアプリを有している場合がある。この場合、React Native 等の開発フレームワークを使用することで、1 つのソースコードから Android と iOS 等の複数の OS に対応するアプリをビルドしていることがある。検証における工数を効率化する目的で、Android と iOS のアプリで同一の処理を行う箇所については、一方の OS のみで検証が十分かを検討することが望まれる。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。

ファームウェア解析に当たって、ファームウェアの抽出可否の確認にはハードウェアを手動で解析する必要があり、費用が高くなる傾向がある。そのため、検証依頼者からファームウェアファイルを別途受領し、ファームウェアの抽出可否の確認はせず、ファームウェアファイルの解析のみを行う場合もある。なお、検証を実

[7] https://www.jpcert.or.jp/pr/stop-password.html, https://www.nisc.go.jp/security-site/handbook/index.html, https://pages.nist.gov/800-63-3/sp800-63b.html, https://github.com/OWASP/ASVS/tree/master/4.0/docs_en など

施するために、検証対象機器を分解することがある。検証対象機器を分解した後に、原状復帰することが困難な可能性がある場合は、事前に検証依頼者へ伝え、検証依頼者からの合意を得た後に検証を実施することが望まれる。

ソースコード解析に当たって、自動化ツールを活用して検出した脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、検出された脆弱性が多く、すべてを網羅的に確認することが現実的でない場合、脆弱性の深刻度や検証の工数等を鑑み、再現性の確認を行う脆弱性を決定することが必要である。

7 スマート家電に関するセキュリティ検証プラクティス

7.1 機器の概要・想定脅威

スマート家電とは、インターネットとの接続機能や専用のスマートフォンアプリと連携した遠隔操作等の機能を有する家電である。具体的には、スマート TV、スマートリモコン、ロボット掃除機、スマートスピーカー等がスマート家電として挙げられる。なお、スマート家電には、遠隔操作を制御するための Web アプリケーションも含めて構築されることがある。一例として、スマート TV の一般的なネットワークの構成及び想定利用環境を図 7-1 に示す。

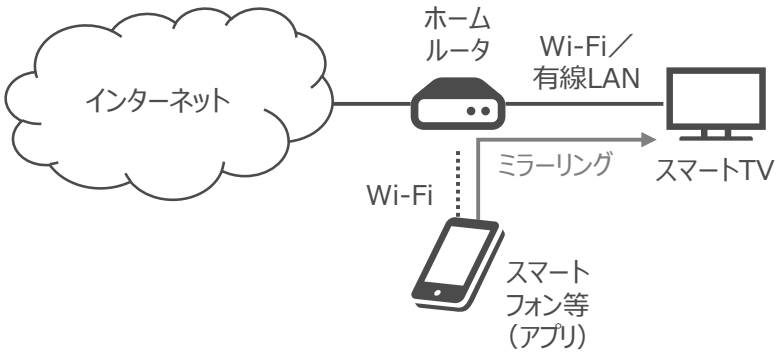

図 7-1 スマート TV の想定利用環境

スマート家電の想定脅威として、攻撃者が不正に遠隔操作することや遠隔操作の内容を改ざんすることが考えられる。例えば、家庭内のロボット掃除機が不正に遠隔操作された場合、宅内の資産に対して物理的な損害を与えられる可能性がある。攻撃手法としては、スマート家電・サーバ間の通信の中間者攻撃による不正な遠隔操作、有線 LAN ・Wi-Fi を介した不正な遠隔操作、サーバのなりすましによる不正な遠隔操作、なりすましたスマートフォンアプリによる不正な遠隔操作、等が考えられる。加えて、スマート TV のように、USB などで接続する外部ストレージが存在する場合、USB 接続の外部ストレージからの情報窃取や不正なアプリがインストールされる脅威が考えられる。他にも、スマート家電を第三者が物理的に攻撃できる場合、正規ユーザが通常行わないような操作によって、スマート家電に不具合を引き起こされる脅威が考えられる。これらのような脆弱性について検証することが望まれる。

7.2 想定される検証環境

スマート家電で想定される検証環境の一例として、スマート TV の検証環境を図 7-2 に示す。検証に当たっては、検証事業者内に検証専用の LAN を構築して検証することが望まれる。なお、スマート家電における脅威は、スマート家電本体のみではないため、遠隔操作等のためのスマートフォンアプリや遠隔操作を制御するための Web アプリケーション等も検証対象とすることが望まれる。

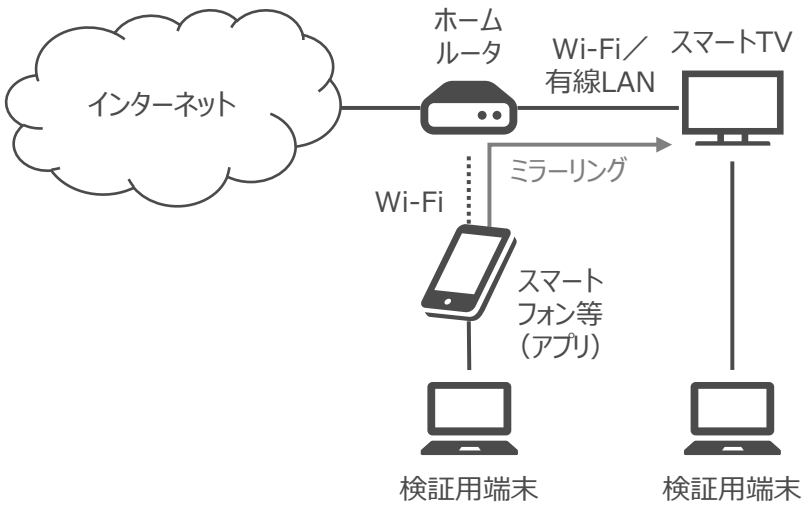

図 7-2 スマート TV に対する想定検証環境

7.3 実証において適用された検証手法

実証におけるスマート家電の検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。

また、遠隔操作等のためのスマートフォンアプリに対しては、静的解析・動的解析のツールを使用することが効果的である。静的解析・動的解析においては、スマート家電の操作が第三者によって可能かを確認することが必要となるため、ペアリング時や遠隔操作時の通信方式の妥当性、処理の不備の有無、利用者認証情報の適切な保護等を確認することが望まれる。攻撃者の視点に立てば、悪用できるアプリに制限はない。そのため、遠隔操作等のためのスマートフォンアプリが、Android や iOS 等の複数の OS で用意されている場合には、複数 OS のアプリに対して解析を行うことが望まれる。Web アプリケーションに対しては、自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。なお、自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

スマート家電や遠隔操作等のためのスマートフォンアプリの通信インタフェースや Web アプリケーションに対して不正なデータを送信し、OS や Web アプリケーションの応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。また、スマート家電本体に対して、電源スイッチの ON/OFF の切り替えを大量に試行する等、正規ユーザが通常行わないような操作によって、スマート家電が停止しないか、不具合が発生しないかを確認することも有効である。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

スマート家電と遠隔操作等のためのスマートフォンアプリ間の通信の盗聴や中間者攻撃の可能性を確認する。スマート家電の遠隔操作のための通信は、有線 LAN、Wi-Fi、Bluetooth で行われることが多く、この通信を盗聴できるかをパケットキャプチャにより検証することが望まれる。遠隔操作のための通信をキャプチャする方法として、スマート家電になりすまし、なりすました機器を正規のスマートフォン等と接続することで通信内容を取得することが考えられる。なりすましの結果、スマート家電の遠隔操作に必要なリクエストを入手できる場合、そのリクエストを悪用し、スマート家電に送信することで、遠隔操作が可能かを検証することができる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（6） ファームウェア解析

検証対象機器のインタフェースを介してファームウェアへのアクセスが可能となる場合があるため、ファームウェアの抽出可否を確認する。仮にファームウェアの抽出が可能であった場合に、当該ファームウェアファイルを解析し、ファイルシステムの取り出し可否を確認する。ファイルシステムの取り出しが可能であった場合に、ファイルの内容を解析し、機微な情報の取得が可能であるか否かを確認する。

検証の詳細手順：手引き別冊 1 第 4.3 節参照

（7） バイナリ解析

抽出したファームウェア等のバイナリファイルに対して、脆弱性の可能性のある箇所を確認する。脆弱性の可能性のある箇所が存在した場合、該当箇所で必ずしも脆弱性が発生するわけではない。そのため、リバースエンジニアリングツールを用いてバイナリファイルをデコンパイルし、ソースコードを確認した上で、当該脆弱性の悪用可能性を確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.4 節参照

（8） ソースコード解析

Web アプリケーションのソースコードを確認し、脆弱性が含まれていないかを静的に解析する。ソースコード解析では、自動化ツールを活用してソースコードに含まれる特定のパターンを抽出することで、脆弱性を検出するほか、また、自動化ツールを使用せず、検証作業者によるマニュアル診断も想定される。ソースコードの確認観点は、機微情報がハードコートされていないか、ビルド時にソースコードが難読化されているか等が挙げられる。なお、ビルド時にソースコードが難読化されていない場合、攻撃者がプログラムを解析する目的でデコンパイルした場合、ソースコードを解析しやすく、この場合、攻撃するためにヒントとなる情報の窃取や検証対象機器への想定していない操作による攻撃が実施される可能性がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。

7.4 実証において検出された代表的な脆弱性

実証事業で検証対象としたスマート家電において検出された代表的な脆弱性は以下のとおりである。

表 7-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	Web アプリケーションにおいて他ユーザの情報変更が可能	悪用されることで、ユーザのメールアドレスやパスワード等の窃取や改ざんが行われる可能性がある。この結果、メールアドレスやパスワード等を変更されたユーザは Web アプリケーションへログインできなくなり、当該スマート家電の遠隔操作やスマート家電の動作が停止するおそれがある。	・ Web アプリケーションへログインし、ユーザの情報変更画面へアクセスする。 ・ ユーザの情報変更画面で、他ユーザで登録されている住所や電話番号等の情報を変更したい内容を記入し、変更ボタンを押下する。変更ボタンを押下した際のリクエストに対して、Burp Suite 等のローカルプロキシツールを使用し、契約番号を他ユーザの契約番号へ書き換えて送信する。 ・ 他ユーザで Web アプリケーションへログインし、ユーザの情報が変更されたことを確認する。
2	adb（android debug bridge）を利用したスマート家電への接続・任意操作可能	悪用されることで、スマート家電で保有しているデータの窃取や改ざん、不正なアプリケーションのインストール等が行われる可能性がある。この結果、当該スマート家電を利用しているユーザの情報漏えい、スマート家電の動作停止等が引き起こされるおそれがある。	・ 検証対象機器に対して、USB 経由にて adb を利用した接続を行い、任意の adb コマンドを入力する。 ・ 入力した操作に対する対象機器のレスポンスを踏まえ、脆弱性の有無を確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
3	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、マルウェアに感染し、スマート家電の遠隔操作やスマート家電の動作が停止するおそれがある。	・ OWASP ZAP のスパイダー機能を用い、検証対象機器の Web アプリケーションにおいてアクセス可能な URL 一覧を作成する。 ・ ファジングの対象とする URL とパラメータのリストを作成する。パラメータリストの作成に当たっては、GitHub 等で公開されているペイロードリスト [8]を参考にする。 ・ 作成したパラメータリストに基づき、ファジングを実施する。 ・ ファズデータに対する Web サービスのレスポンスを踏まえ、脆弱性の有無を判断する。
4	Web アプリケーションにおけるファイルアップロード機能において任意のファイルをアップロード可能	悪用されることで、任意のコードが実行される可能性がある。この結果、マルウェアに感染し、スマート家電の遠隔操作やスマート家電の動作が停止するおそれがある。	・ Web アプリケーションにおけるファイルアップロード機能において、開発者が意図しない任意のコードを含むファイルアップロードが可能かを試行するため、検証用のファイルを作成する。 ・ 作成した検証用ファイルをアップロード機能においてアップロードする。 ・ 検証用ファイルをアップロードしたことに対する Web アプリケーションのレスポンスを踏まえ、脆弱性の有無を判断する。

[8] https://github.com/payloadbox, https://github.com/xmendez など

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
5	Web アプリケーションにおけるバッファオーバーフローの脆弱性	悪用されることで、メモリ上の値が不正に書き換えられ、任意のコードが実行される可能性がある。この結果、マルウェアに感染し、スマート家電の遠隔操作やスマート家電の動作が停止するおそれがある。	・ 検証対象となるソースコードに対して、ソースコード解析ツールを用いた静的解析を行い、脆弱性の可能性のある箇所を特定する。 ・ 特定した脆弱性の可能性のある箇所に対して、脆弱性の悪用可能性を確認する。

7.5 想定される推奨事項

表 7-1 で示した、実証事業で検証対象としたスマート家電において検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 7-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	Web アプリケーションにおいて他ユーザの情報変更が可能	・ ユーザの登録情報変更処理時に、送信リクエスト内容に含まれる契約番号によって処理の対象となる契約情報を指定しないようにする。
2	adb を利用したスマート家電への接続・任意操作可能	・ adb の利用を想定していないスマート家電の場合は、adbd（adb の DAEMON）を無効にする。 ・ adb の利用を想定しているスマート家電の場合は、adb を利用して接続された際に認証処理を行う。なお、adbd には、USB 経由で接続した端末が想定した端末であるかを認証するための機能を有するため、この機能を利用し対策することも可能である。
3	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	・ シェルを起動できる言語機能の使用を避ける。 ・ 引数に対してチェックを行い、あらかじめ許可した処理のみ実行する。
4	Web アプリケーションにおけるファイルアップロード機能において任意のファイルをアップロード可能	・ ユーザがアップロードするファイルを検証し、アップロード可能かを確認する処理を行う。

項番	脆弱性の概要	推奨される対策事項
5	Web アプリケーションにおけるバッファオーバーフローの脆弱性	・ バッファのサイズをチェックする処理をコードに追加する。

7.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、スマート家電の機能やアーキテクチャを調査・分析することが望まれる。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、スマート家電がどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。遠隔操作等のためのスマートフォンアプリに関しては、Android と iOS 等の複数の OS に対応するアプリを有している場合がある。この場合、React Native 等の開発フレームワークを使用することで、1 つのソースコードから Android と iOS 等の複数の OS に対応するアプリをビルドしていることがある。検証における工数を効率化する目的で、Android と iOS のアプリで同一の処理を行う箇所については、一方の OS のみで検証が十分かを検討することが望まれる。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。

ファームウェア解析に当たって、ファームウェアの抽出可否の確認にはハードウェアを手動で解析する必要があり、費用が高くなる傾向がある。そのため、検証依頼者からファームウェアファイルを別途受領し、ファームウェアの抽出可否の確認はせず、ファームウェアファイルの解析のみを行う場合もある。なお、検証を実施するために、検証対象機器を分解することがある。検証対象機器を分解した後に、原状復帰することが困難な可能性がある場合は、事前に検証依頼者へ伝え、検証依頼者からの合意を得た後に検証を実施することが望まれる。

バイナリ解析に当たって、抽出・受領したファームウェア等のバイナリファイルについて、静的解析ツールを用い、脆弱性の可能性のある箇所を特定するだけでなく、特定した箇所の脆弱性の再現性（悪用可能性）を確認することが必要である。脆弱性の再現性（悪用可能性）を確認するためにも、適宜、リバースエンジニアリングツールを用い、バイナリファイルをデコンパイルし、コードを確認することが必要である。

ソースコード解析に当たって、自動化ツールを活用して検出した脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、検出された脆弱性が多く、すべてを網羅的に確認することが現実的でない場合、脆弱性の深刻度や検証の工数等を鑑み、再現性の確認を行う脆弱性を決定することが必要である。

8 ドローンに関するセキュリティ検証プラクティス

8.1 機器の概要・想定脅威

航空法では、100g 以上の人が乗ることができない飛行機、回転翼航空機、滑空機、飛行船のうち、遠隔操作又は自動操縦により飛行させることができるものが「無人航空機」として定義され、機体重量が 100g 未満の航空機（一般的に「トイドローン」と呼ばれる）は模型航空機に分類される。本章では、特に複数の回転翼を有したマルチコプターと呼ばれる無人航空機のうち、主に空撮に用いられる航空機をドローンと呼ぶ。ドローンに関するシステムは、ドローン本体に加えて、ドローンを操縦するためのコントローラ、フライト計画・機体管理するためのスマートフォンアプリ等のアプリケーション、撮影した写真や動画を格納するためのサーバで構成されることが多い。一般的なドローンに関するシステム構成を図 8-1 に示す。

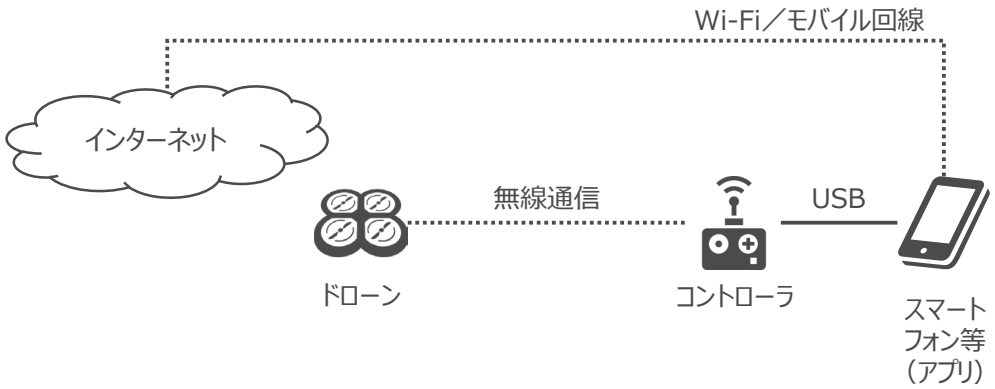

図 8-1 ドローンの想定システム構成

ドローン特有の想定脅威として、ドローンを攻撃者が不正に操作し、ドローン自体や周辺の資産に対して物理的な損害を与えることが考えられる。また、フライトログデータ等の重要データを窃取する脅威も想定される。前者の脅威については、なりすましたコントローラやスマートフォンアプリ等のアプリケーションによる不正操作が攻撃手法として考えられる。後者の脅威については、スマートフォンアプリ等のアプリケーションとサーバ間の通信の中間者攻撃による情報窃取が考えられる。これらの脅威や攻撃手法を実現しうる脆弱性の有無を確認するため、コントローラやスマートフォンアプリ等のアプリケーションに対し、既知の攻撃手法に対する影響有無の確認や、通信インタフェースに関する脅威に対する検証が求められる。

8.2 想定される検証環境

ドローンの検証に当たって想定される検証環境は図 8-2 に示すとおりであり、検証事業者内に検証専用の LAN を構築して検証することが望まれる。なお、ドローンにおける脅威は、ドローン本体のみではないため、コントローラやスマートフォンアプリ等のアプリケーションも検証対象とすることが望まれる。

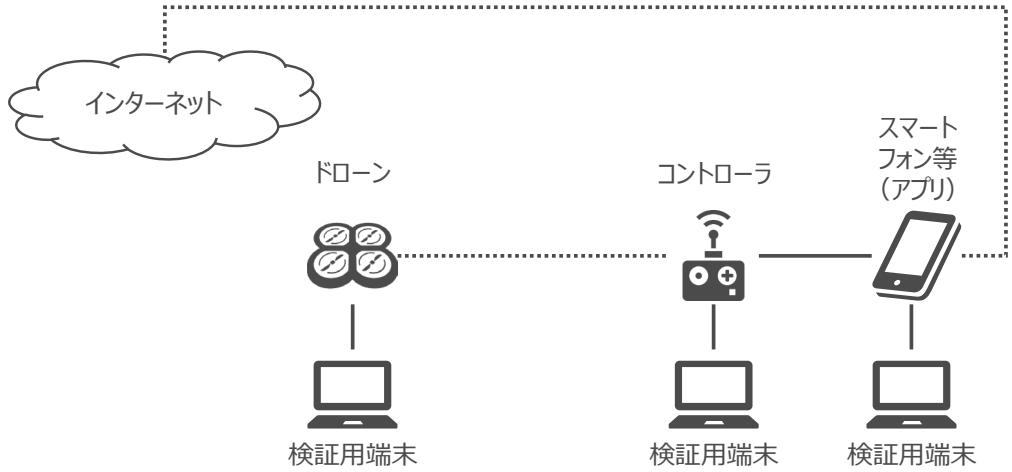

図 8-2 ドローンに対する想定検証環境

8.3 実証において適用された検証手法

実証におけるドローンの検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） 既知脆弱性の診断

検証対象機器に対して、既知の脆弱性が存在しないかを確認する。コントローラやスマートフォンアプリ等のアプリケーションの Web インタフェースに対して自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。特に、ドローンの脅威となるユーザの権限設定について、適切に実装されているか、不正なペアリング実装方式となっていないかを確認することが望まれる。自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（3）　ファジング

検証対象機器の通信インタフェースや Web インタフェースに対して不正なデータを送信し、アプリケーションや OS の応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。機能停止に対する防御性能を確認するためには、フラッド攻撃をはじめとする機器のリソースを消費される攻撃手法を試行し、サービス不能状態に陥ることがないかを確認することが有効である。ファズデータの網羅性を担保することは困難であるため、検証範囲を絞るために、ファジング対象のインタフェースやファズデータのパラメータ数について、事前に検証依頼者と相談することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（4）　ネットワークキャプチャ

検証対象機器の通信内容を取得し、平文の通信、暗号強度の低い通信、想定外の通信先への通信の有無を確認する。検証対象範囲について、最低限 Ethernet の通信に関するキャプチャを行うことが望まれる。また、ネットワークキャプチャの対象時間について、最低限 60 分程度の通信をキャプチャし、通信先・通信内容を確認することが望まれる。ドローン本体とコントローラ間の通信に汎用プロトコルを用いている場合、当該プロトコルに対する中間者攻撃の可能性を確認することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（5）　ソースコード解析

スマートフォンアプリ等のアプリケーションのソースコードを確認し、脆弱性が含まれていないかを静的に解析する。ソースコード解析では、自動化ツールを活用してソースコードに含まれる特定のパターンを抽出することで、脆弱性を検出するほか、また、自動化ツールを使用せず、検証作業者によるマニュアル診断も想定される。ソースコードの確認観点は、機微情報がハードコートされていないか、ビルド時にソースコードが難読化されているか等が挙げられる。なお、ビルド時にソースコードが難読化されていない場合、攻撃者がプログラムを解析する目的でデコンパイルした場合、ソースコードを解析しやすく、この場合、攻撃するためにヒントとなる情報の窃取や検証対象機器への想定していない操作による攻撃が実施される可能性がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。

8.4 実証において検出された代表的な脆弱性

実証事業で検証対象としたドローンにおいて検出された代表的な脆弱性は以下のとおりである。

表 8-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	ドローン本体のネットワーク設定機能における OS コマンドインジェクションの脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、ドローンの動作が停止し、当該ドローンを使用している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ 検証対象となるソースコードに対して、ソースコード解析ツールを用いた静的解析を行い、脆弱性の可能性のある箇所を特定する。 ・ 特定した脆弱性の可能性のある箇所に対して、脆弱性の悪用可能性を確認する。
2	Web アプリケーションにおけるファイルアップロード機能において任意のコードを実行可能	悪用されることで、任意のコードが実行される可能性がある。この結果、ドローンの動作が停止し、当該ドローンを使用している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Web アプリケーションにおけるファイルアップロード機能において、開発者が意図しない任意のコードを含むファイルアップロードが可能かを試行するため、検証用のファイルを作成する。 ・ 作成した検証用ファイルをアップロード機能においてアップロードする。 ・ 検証用ファイルをアップロードしたことに対する Web アプリケーションのレスポンスを踏まえ、脆弱性の有無を判断する。
3	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、ドローンの動作が停止し、当該ドローンを使用している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ 検証対象となるソースコードに対して、ソースコード解析ツールを用いた静的解析を行い、脆弱性の可能性のある箇所を特定する。 ・ 特定した脆弱性の可能性のある箇所に対して、脆弱性の悪用可能性を確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
4	公開フォルダ上に機密情報が設置	外部の攻撃者によってソースコード等の機密情報を取得されることによる情報漏えいの可能性がある。ソースコードを取得・解析され、脆弱性を発見された場合、脆弱性を悪用された攻撃が行われるおそれがある。	・ 検証対象となるドローンに関連する資料等の情報が設置された公開フォルダへアクセスする。 ・ 公開フォルダに設置されている資料等の情報を取得する。 ・ 取得した資料等の情報の中に、ソースコード等の機密情報が含まれているかを確認する。

8.5 想定される推奨事項

表 8-1 で示した、実証事業で検証対象としたドローンにおいて検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 8-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	ドローン本体のネットワーク設定機能における OS コマンドインジェクションの脆弱性	・ 入力された文字列をすべて許容する関数の使用を避ける。 ・ 入力された文字列に対してチェックを行い、あらかじめ許可した文字列のみを許容する。
2	Web アプリケーションにおけるファイルアップロード機能において任意のコードを実行可能	・ ユーザがアップロードするファイルを検証し、アップロード可能かを確認する処理を行う。
3	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	・ シェルを起動できる言語機能の使用を避ける。 ・ 引数に対してチェックを行い、あらかじめ許可した処理のみ実行する。
4	公開フォルダ上に機密情報が設置	・ 公開フォルダ上に設置する情報の範囲を適切に設定する。

8.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、ドローンの機能やアーキテクチャを調査・分析することが望まれる。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、ドローンがどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。

既知脆弱性の診断に当たって、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。例えば、HTTP サーバに対する検証を行う場合は、HTTP におけるファズデータ（例：膨大な長さの URL）を生成する必要があり、HTTP サーバが処理しないレイヤのプロトコル（例：TCP/IP）におけるファズデータを生成しても有効な検証とはならない。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。

また、スマートフォン等のアプリケーションとサーバが接続しているケースが考えられる。本手引きでは、サーバを対象とする検証は対象外としているが、サーバにおいては、悪意ある第三者によってデータの改ざんや搾取、機能を停止される等の脅威も想定される。もし、検証依頼者の依頼を踏まえサーバに対する検証を行う場合、サーバへの通信方式の妥当性、フライトログデータの保管方法の妥当性、利用者認証情報の適切な保護等を確認することが望ましい。なお、稼働中のサーバに対して検証を行う場合、他のサービスに影響を与えないか等を事前に検証依頼者と協議し、双方の合意の下で検証を実施することが必要である。

ソースコード解析に当たって、自動化ツールを活用して検出した脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、検出された脆弱性が多く、すべてを網羅的に確認することが現実的でない場合、脆弱性の深刻度や検証の工数等を鑑み、再現性の確認を行う脆弱性を決定することが必要である。

9 ネットワークカメラに関するセキュリティ検証プラクティス

9.1 機器の概要・想定脅威

ネットワークカメラとは、インターネットと接続機能を有するカメラである。撮影した映像データをインターネット経由でクラウドサーバ上へ保存し、専用のスマートフォン等の Web アプリケーションで映像データを確認することができる。また、Web アプリケーションでは、映像の確認だけでなく、撮影方向の変更やズーム等の操作を行うことができる。一例として、ネットワークの構成及びネットワークカメラの想定利用環境を図 9-1 に示す。

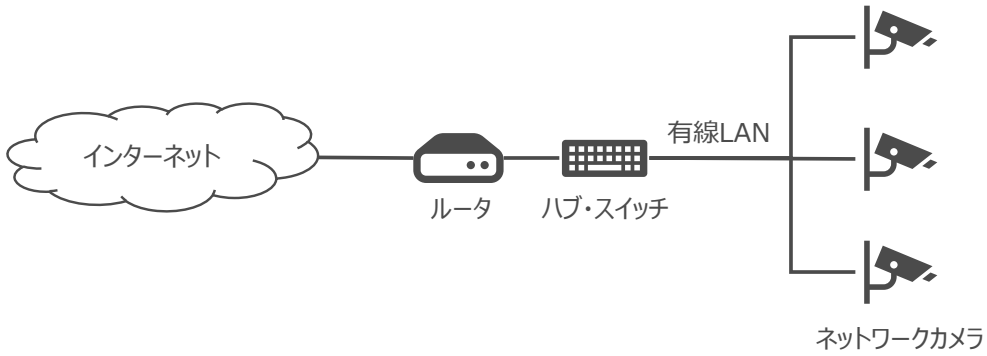

図 9-1 ネットワークカメラの想定利用環境

ネットワークカメラ特有の想定脅威として、ネットワークカメラで撮影した映像データを窃取・削除・改ざんされることが考えられる。クラウドサーバ上にデータが保存されている場合、データを保存する際の通信データの盗聴、Web アプリケーションへの不正なログインが攻撃手法として考えられる。また、ネットワークカメラを屋外に設置している等でネットワークカメラ本体への物理的な攻撃の脅威として、SD カード等の外部ストレージからアプリがインストールされ、マルウェアに感染する脅威もある。これらの脅威や攻撃手法を実現しうる脆弱性の有無を確認するため、ネットワークカメラ本体や専用のスマートフォン等の Web アプリケーションに対する既知の攻撃手法に対する影響の確認や、通信インタフェースに関する脅威に対する検証が求められる。

9.2 想定される検証環境

ネットワークカメラの検証に当たって想定される検証環境は図 9-2 に示すとおりであり、検証事業者内に検証専用の LAN を構築して検証することが望まれる。なお、ネットワークカメラにおける脅威は、ネットワークカメラ本体のみではないため、ネットワークカメラを操作するための Web アプリケーションも検証対象とすることが望まれる。

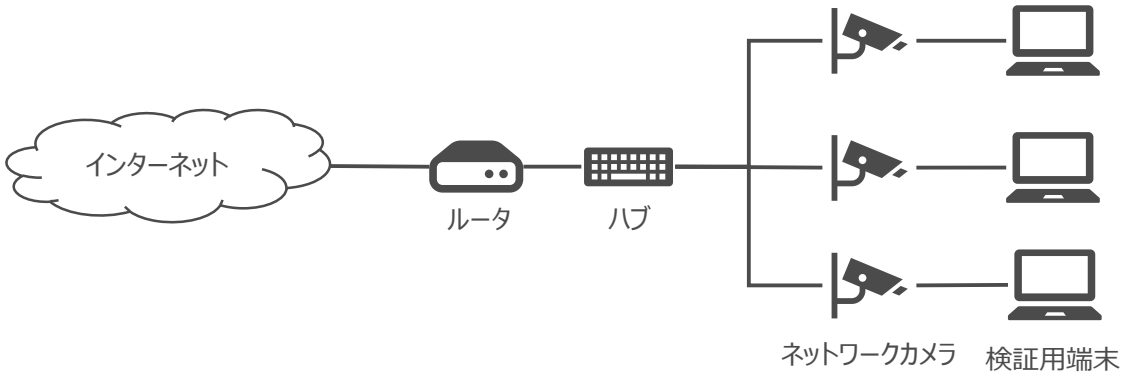

図 9-2 ネットワークカメラに対する想定検証環境

9.3 実証において適用された検証手法

実証におけるネットワークカメラの検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。また、Web アプリケーションに対して自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能であ

るか、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

検証対象機器の通信インタフェースや Web アプリケーションに対して不正なデータを送信し、アプリケーションや OS の応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。機能停止に対する防御性能を確認するためには、フラッド攻撃をはじめとする機器のリソースを消費される攻撃手法を試行し、サービス不能状態に陥ることがないかを確認することが有効である。また、Web アプリケーションに対しては、ファジングツールを用いたスキャンで検証を自動化することが有効である。ただし、ファジングツールにより検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

検証対象機器の通信内容を取得し、平文の通信、暗号強度の低い通信、想定外の通信先への通信の有無を確認する。検証対象範囲について、最低限 Ethernet の通信に関するキャプチャを行うことが望まれる。また、ネットワークキャプチャの対象時間について、最低限 60 分程度の通信をキャプチャし、通信先・通信内容を確認することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（6） ファームウェア解析

検証対象機器のインタフェースを介してファームウェアへのアクセスが可能となる場合があるため、ファームウェアの抽出可否を確認する。仮にファームウェアの抽出が可能であった場合に、当該ファームウェアファイルを解析し、ファイルシステムの取り出し可否を確認する。ファイルシステムの取り出しが可能であった場合に、ファイルの内容を解析し、機微な情報の取得が可能であるか否かを確認する。

検証の詳細手順：手引き別冊 1 第 4.3 節参照

（7）　バイナリ解析

抽出したファームウェア等のバイナリファイルに対して、脆弱性の可能性のある箇所を確認する。脆弱性の可能性のある箇所が存在した場合、該当箇所で必ずしも脆弱性が発生するわけではない。そのため、リバースエンジニアリングツールを用いてバイナリファイルをデコンパイルし、ソースコードを確認した上で、当該脆弱性の悪用可能性を確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.4 節参照

9.4 実証において検出された代表的な脆弱性

実証事業で検証対象としたネットワークカメラにおいて検出された代表的な脆弱性は以下のとおりである。

表 9-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、マルウェアに感染し、ネットワークカメラの不正操作やネットワークカメラの動作の停止を行われるおそれがある。	・ Web アプリケーションへログインし、検証対象機器との疎通確認のための画面へアクセスする。 ・ 検証対象機器の情報を入力し、疎通確認実行ボタンを押下する。疎通確認実行ボタンを押下した際の HTTP リクエストに対して、Burp Suite 等のローカルプロキシツールを使用し、任意の OS コマンドを追加して送信する。 ・ 任意の OS コマンドが実行されたことを確認する。
2	Web アプリケーションおける認証制御の不備	悪用されることで、Web アプリケーションへ不正にログインされる可能性がある。この結果、当該ネットワークカメラで撮影したデータの窃取、ネットワークカメラの動作の停止を行われるおそれがある。	・ Web アプリケーションのログイン画面へアクセスする。 ・ ログイン画面で受け取った認証情報入力のリクエストデータをローカルプロキシツールで取得する。 ・ 取得したリクエストデータに対して、Host ヘッダの値を localhost へ変更して認証サーバへ返信する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
3	Web アプリケーションにおけるクロスサイトリクエストフォージェリの脆弱性	悪用されることで、正規ユーザの意図しない操作を行われる可能性がある。この結果、Web アプリケーションへのログイン ID やパスワードの変更、当該ネットワークカメラの動作の停止を行われるおそれがある。	・ 登録ボタンを押下することで、Web アプリケーション内でネットワークカメラの設定をデフォルトに戻して再起動する動作と同様の処理を行う罠ページを準備する。 ・ Web アプリケーションへログインし、準備した罠ページへアクセスし、登録ボタンを押下する。 ・ 罠ページでの登録ボタン押下後のレスポンスを踏まえ、脆弱性の有無を判断する。
4	不正な SD カードを挿入することによる任意コード実行の脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、ネットワークカメラの動作の停止、ネットワークカメラに関する重要情報の漏えいを行われるおそれがある。	・ 実行したい任意のコードを含むファイルを作成し、特定のファイル名としてルート階層に保存した SD カードを準備する。 ・ 準備した SD カードを検証対象機器に挿入して起動する。 ・ 検証対象機器や SD カードのレスポンスを踏まえ、脆弱性の有無を判断する。
5	Web アプリケーションへのログインパスワードを不正取得可能	悪用されることで、Web アプリケーションへ不正にログインされる可能性がある。この結果、当該ネットワークカメラ内に保存されたデータの窃取、ネットワークカメラの動作の停止を行われるおそれがある。	・ Web アプリケーション内の特定 URL へアクセスする。 ・ アクセスしたページに表示されているパスワードを確認する。 ・ 確認したパスワードを使用して Web アプリケーションへログインする。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
6	アップデート用ファームウェアファイルから重要情報取得可能	悪用されることで、ネットワークカメラのアップデート機能や Web アプリケーションの認証情報等の重要情報を取得される可能性がある。この結果、ネットワークカメラの動作の停止、Web アプリケーションへの不正アクセスが行われるおそれがある。	・ 検証対象となるファームウェアファイルを binwalk 等のファームウェア解析ツールを使用して解析する。 ・ 解析した結果を踏まえ、脆弱性の有無を判断する。

9.5 想定される推奨事項

表 9-1 で示した、実証事業で検証対象としたネットワークカメラにおいて検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 9-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	・ シェルを起動できる言語機能の使用を避ける。 ・ 引数に対してチェックを行い、あらかじめ許可した処理のみ実行する。
2	Web アプリケーションおける認証制御の不備	・ 認証において、Authorization ヘッダの情報のみを使用する。変更が可能なその他のヘッダの値を使用しない。 ・ 認証方式を Basic 認証にし、通信プロトコルを HTTPS に変更する。
3	Web アプリケーションにおけるクロスサイトリクエストフォージェリの脆弱性	・ 正規ユーザからネットワークカメラの設定に関する機能を実行する URL にアクセスがあった場合、サーバ側でログインしたユーザによる正規の操作であるかのチェックを行う。
4	不正な SD カードを挿入することによる任意コード実行の脆弱性	・ SD カードから読み込んだデータを展開・実行する機能を削除する。なお、当該機能の削除が困難な場合、読み込んだデータに対して署名を施し、展開・実行時に署名検証を行う。

項番	脆弱性の概要	推奨される対策事項
5	Web アプリケーションへのログインパスワードを不正取得可能	・ Web アプリケーションへのアクセス時の認証処理をネットワークカメラ側で実施することで、適切な認証処理を行う。
6	アップデート用ファームウェアファイルから重要情報取得可能	・ 暗号化を実施し、暗号鍵を特定できない場所に保管する。

9.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、ネットワークカメラの機能やアーキテクチャを調査・分析することが望まれる。例えば、屋外に設置するネットワークカメラの場合、POE ハブによって電源を供給することがあり、POE ハブのインタフェースを経由した攻撃が想定されると分析される。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、ネットワークカメラがどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。また、ファジングツールを用いたスキャンで検証を実施した場合、検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望まれる。

ファームウェア解析に当たって、ファームウェアの抽出可否の確認にはハードウェアを手動で解析する必要があり、費用が高くなる傾向がある。そのため、検証依頼者からファームウェアファイルを別途受領し、ファームウェアの抽出可否の確認はせず、ファームウェアファイルの解析のみを行う場合もある。なお、検証を実施するために、検証対象機器を分解することがある。検証対象機器を分解した後に、原状復帰することが困難な可能性がある場合は、事前に検証依頼者へ伝え、検証依頼者からの合意を得た後に検証を実施することが望まれる。

バイナリ解析に当たって、抽出・受領したファームウェア等のバイナリファイルについて、静的解析ツールを用い、脆弱性の可能性のある箇所を特定するだけでなく、特定した箇所の脆弱性の再現性（悪用可能性）を確認することが必要である。脆弱性の再現性（悪用可能性）を確認するためにも、適宜、リバースエンジニアリングツールを用い、バイナリファイルをデコンパイルし、コードを確認することが必要である。

ネットワークカメラは撮影したデータをクラウドサーバと接続し、保存しているケースが考えられる。本手引きでは、クラウドサーバを対象とする検証は対象外としているが、クラウドサーバにおいては、悪意ある第三

者によってデータの改ざんや搾取、機能を停止される等の脅威も想定される。もし、検証依頼者の依頼を踏まえクラウドサーバに対する検証を行う場合、通信インタフェースやクラウドサーバの管理コンソールに対する検証を実施することが望ましい。なお、稼働中のクラウドサーバに対して検証を行う場合、他のサービスに影響を与えないか等を事前に検証依頼者と協議し、双方の合意の下で検証を実施することが必要である。

10 センサ・監視装置に関するセキュリティ検証プラクティス

10.1 機器の概要・想定脅威

センサ・監視装置とは、主に産業用として使用され、工場設備や工作機械等の監視対象物を監視し、監視対象物の映像、音、温度等を検知し、稼働状況や異常データを取得する機器やシステムである。取得したデータは、クラウドサーバに蓄積され、PC 等の監視用端末の Web アプリケーションで確認し、工場設備の最適化や工作機械の稼働効率化等を図るために活用される。一例として、ネットワークの構成及びセンサ・監視装置の想定利用環境を図 10-1 に示す。

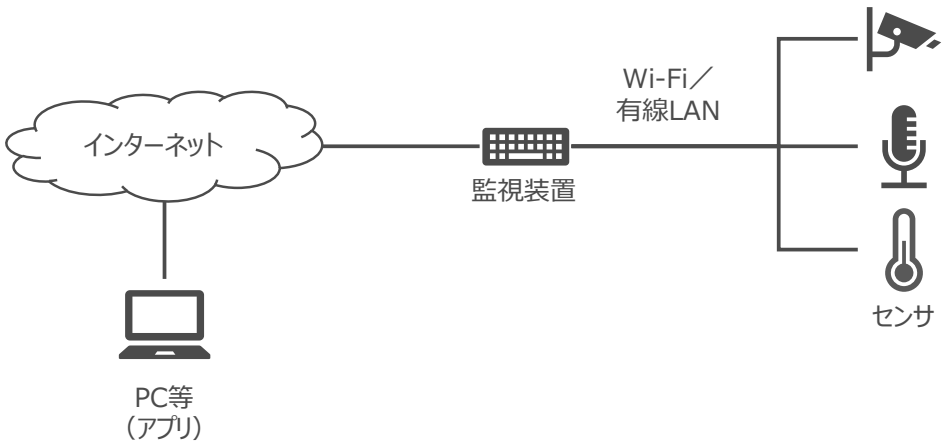

図 10-1 センサ・監視装置の想定利用環境

センサ・監視装置の想定脅威として、センサ・監視装置で取得したデータを窃取・削除・改ざんされることが考えられる。クラウドサーバ上にデータが保存されている場合、データを保存する際の通信データの盗聴、Web アプリケーションへの不正なログインが攻撃手法として考えられる。センサ・監視装置本体にデータが保存されている場合、センサ・監視装置本体に不正に侵入し、データを窃取する攻撃手法が考えられる。また、センサ・監視装置本体への物理的な攻撃の脅威として、SD カード等の外部ストレージからアプリがインストールされ、マルウェアに感染する脅威もある。これらの脅威や攻撃手法を実現しうる脆弱性の有無を確認するため、センサ・監視装置本体や Web アプリケーションに対する既知の攻撃手法に対する影響の確認や、通信インタフェースに関する脅威に対する検証が求められる。

10.2 想定される検証環境

センサ・監視装置の検証に当たって想定される検証環境は図 10-2 に示すとおりであり、検証事業者内に検証専用の LAN を構築して検証することが望まれる。なお、センサ・監視装置における脅威は、センサ・監視装置本体のみではないため、センサ・監視装置を操作・管理するための Web アプリケーションも検証対象とすることが望まれる。

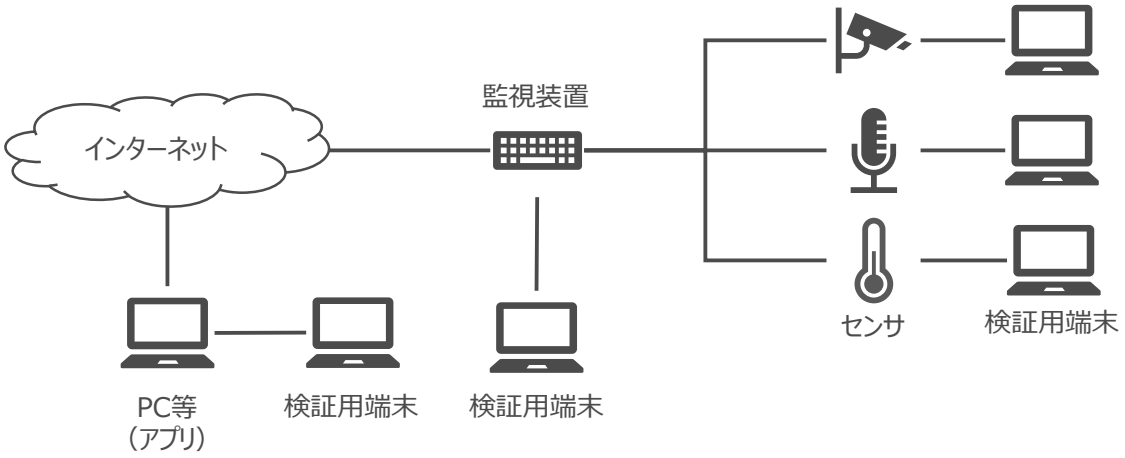

図 10-2 センサ・監視装置に対する想定検証環境

10.3 実証において適用された検証手法

実証におけるセンサ・監視装置の検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。また、Web アプリケーションに対して自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対

する影響がないかを確認する。自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

検証対象機器の通信インタフェースや Web アプリケーションに対して不正なデータを送信し、アプリケーションや OS の応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。機能停止に対する防御性能を確認するためには、フラッド攻撃をはじめとする機器のリソースを消費される攻撃手法を試行し、サービス不能状態に陥ることがないかを確認することが有効である。また、検証対象機器が SD カード等の外部ストレージに関するインタフェースを有している場合、不正なファイルを介したファイルファジングによって応答を確認することも望まれる。Web アプリケーションに対しては、ファジングツールを用いたスキャンで検証を自動化することが有効である。ただし、ファジングツールにより検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

検証対象機器の通信内容を取得し、平文の通信、暗号強度の低い通信、想定外の通信先への通信の有無を確認する。検証対象範囲について、最低限 Ethernet の通信に関するキャプチャを行うことが望まれる。また、ネットワークキャプチャの対象時間について、最低限 60 分程度の通信をキャプチャし、通信先・通信内容を確認することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（6） ファームウェア解析

検証対象機器のインタフェースを介してファームウェアへのアクセスが可能となる場合があるため、ファームウェアの抽出可否を確認する。仮にファームウェアの抽出が可能であった場合に、当該ファームウェアファイルを解析し、ファイルシステムの取り出し可否を確認する。ファイルシステムの取り出しが可能であった場合に、ファイルの内容を解析し、機微な情報の取得が可能であるか否かを確認する。また、OS、ファームウェ

ア、ファイルシステムが SD カード等に格納されている場合、SD カード等に対する物理的アクセス可能性を確認するほか、SD カード等に格納されたデータが暗号化されているかを確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.3 節参照

（7） バイナリ解析

抽出したファームウェア等のバイナリファイルに対して、脆弱性の可能性のある箇所を確認する。脆弱性の可能性のある箇所が存在した場合、該当箇所で必ずしも脆弱性が発生するわけではない。そのため、リバースエンジニアリングツールを用いてバイナリファイルをデコンパイルし、ソースコードを確認した上で、当該脆弱性の悪用可能性を確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.4 節参照

（8） ソースコード解析

センサ・監視装置のソースコードを確認し、脆弱性が含まれていないかを静的に解析する。ソースコード解析では、自動化ツールを活用してソースコードに含まれる特定のパターンを抽出することで、脆弱性を検出するほか、また、自動化ツールを使用せず、検証作業者によるマニュアル診断も想定される。ソースコードの確認観点は、機微情報がハードコートされていないか、ビルド時にソースコードが難読化されているか等が挙げられる。なお、ビルド時にソースコードが難読化されていない場合、攻撃者がプログラムを解析する目的でデコンパイルした場合、ソースコードを解析しやすく、この場合、攻撃するためにヒントとなる情報の窃取や検証対象機器への想定していない操作による攻撃が実施される可能性がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。

10.4 実証において検出された代表的な脆弱性

実証事業で検証対象としたセンサ・監視装置において検出された代表的な脆弱性は以下のとおりである。

表 10-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	古いバージョンの dnsmasq におけるバッファオーバーフローの脆弱性	悪用されることで、細工された不正なリクエストを受け、サービス拒否や任意のコードを実行される可能性がある。この結果、センサ・監視装置の動作が停止し、当該センサ・監視装置を導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
2	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、センサ・監視装置の動作が停止し、当該センサ・監視装置を導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Web アプリケーションへログインし、ネットワーク設定画面へのアクセスを試行する。アクセス時に使用する URL に含まれる GET パラメータに任意の OS コマンドを追加する。 ・ 任意の OS コマンドが実行されたことを確認する。
3	DLL ファイルにおける既知のディレクトリトラバーサルの脆弱性	悪用されることで、任意のファイルが読み込まれ、任意のコードを書き込まれる可能性がある。この結果、ファイルに格納された重要情報の漏えい、センサ・監視装置の動作が停止するおそれがある。	・ 検証対象機器の関連アプリケーションをインストールし、インストール時に生成された EXE 及び DLL ファイルを取得する。 ・ 取得したファイル名及びバージョン情報に基づき、既知の脆弱性が公開されていないかを NIST NVD、GitHub、Web 検索、JVN 等を用いて確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
4	管理者権限ユーザの作成における認証回避の脆弱性	悪用されることで、管理者権限ユーザを任意に作成され、任意の操作が実行される可能性がある。この結果、センサ・監視装置の動作が停止し、当該センサ・監視装置を導入している企業の業務や導入しているシステムの運用を停止するおそれがある。	・ Web アプリケーションへログインする。 ・ 特定の文字列から始まるユーザ名で、新たユーザを作成するリクエストを送信する。 ・ 作成したユーザで、Web アプリケーションへログインし、管理者権限が必要な機能を実行する。
5	Web アプリケーションにおける任意コード実行の脆弱性	悪用されることで、任意のコードが実行される可能性がある。この結果、センサ・監視装置の動作が停止し、当該センサ・監視装置を導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Web アプリケーションへログインする。 ・ ユーザからの入力を要求する機能にて、実行したい任意のコードを入力し、機能実行のためのリクエストを送信する。 ・ Web アプリケーションのレスポンスを踏まえ、脆弱性の有無を判断する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
6	OS におけるアカウント認証情報を推測可能	悪用されることで、管理者権限を持つユーザとして不正ログインされる可能性がある。この結果、任意のコマンドを実行し、センサ・監視装置の動作が停止し、当該センサ・監視装置を導入している企業の業務や導入しているシステムの運用を停止するおそれがある。	・ 検証対象機器に挿入されている SD カードを取り出す。 ・ fdisk コマンドを使用して、取り出した SD カードのパーティションテーブルを確認し、検証対象機器で使用されている OS システムが格納されたパーティションを特定する。 ・ 特定した OS システムが格納されたパーティションに対して、マウントを実行し SD カードを読込・利用可能な状態とする。 ・ SD カードに保存されたファイルを確認し、OS のログイン ID を取得する。 ・ 取得したログイン ID とログイン ID から推測されるパスワード（例：ログイン ID と同一のパスワード）を使用してログインができることを確認する。
7	古いバージョンの Grails が使用している外部ライブラリにおける XML 外部エンティティ参照（XXE）の脆弱性	悪用されることで、Web アプリケーション内の任意のファイルにアクセスされる可能性がある。この結果、任意のファイルに格納された重要情報が漏えいするおそれがある。	・ 検証対象となるソースコードに対して、CodeNarc 等のソースコード解析ツールを用いた静的解析を行い、脆弱性の可能性のある箇所を特定する。 ・ 特定した脆弱性の可能性のある箇所に対して、ソースコード解析ツールによる誤検知がないかを目視で確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
8	Web アプリケーションにおけるファイルアップロード機能において任意のファイルをアップロード可能	悪用されることで、任意のコードが実行される可能性がある。この結果、センサ・監視装置の動作が停止し、当該センサ・監視装置を導入している企業の業務や導入しているシステムの運用を停止するおそれがある。	・ Web アプリケーションにおけるファイルアップロード機能において、開発者が意図しない任意のコードを含むファイルアップロードが可能かを試行するため、検証用のファイルを作成する。 ・ 作成した検証用ファイルをアップロード機能においてアップロードする。 ・ 検証用ファイルをアップロードしたことに対する Web アプリケーションのレスポンスを踏まえ、脆弱性の有無を判断する。
9	開放されたポートを経由した不正接続	悪用されることで、攻撃者に不正に接続される可能性がある。センサ・監視装置への接続後、任意のコマンドを実行、マルウェアに感染するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で開放されているポートを特定する。 ・ 特定したポートを経由し、検証対象機器へ接続する。

10.5 想定される推奨事項

表 10-1 で示した、実証事業で検証対象としたセンサ・監視装置において検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 10-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	古いバージョンの dnsmasq におけるバッファオーバーフローの脆弱性	・ 最新の dnsmasq にバージョンアップする。
2	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	・ シェルを起動できる言語機能の使用を避ける。 ・ 引数に対してチェックを行い、あらかじめ許可した処理のみ実行する。
3	DLL ファイルにおける既知のディレクトリトラバーサルの脆弱性	・ 最新の ICSharpCode.SharpZipLib.dll にバージョンアップする。

項番	脆弱性の概要	推奨される対策事項
4	管理者権限ユーザの作成における認証回避の脆弱性	・ 新規ユーザ作成時のリクエストデータ内のユーザ名に対するチェックにて、前方一致ではなく、完全一致でチェックする。
5	Web アプリケーションにおける任意コード実行の脆弱性	・ 受け取ったリクエストに対してチェックを行い、あらかじめ許可した処理のみ実行する。
6	OS におけるアカウント認証情報を推測可能	・ ログイン ID から推測が困難なパスワードを設定する。また、パスワードは機器ごとに異なるパスワードを設定する。
7	古いバージョンの Grails が使用している外部ライブラリにおける XML 外部エンティティ参照（XXE）の脆弱性	・ 最新の Grails にバージョンアップする。
8	Web アプリケーションにおけるファイルアップロード機能において任意のファイルをアップロード可能	・ ユーザがアップロードするファイルを検証し、アップロード可能かを確認する処理を行う。
9	開放されたポートを経由した不正接続	・ 使用していないポートを閉じ、ポートの開放は必要最低限とする。

10.6 検証に当たっての留意事項

検証サービス事業者は、検証の前段階として、センサ・監視装置の機能やアーキテクチャを調査・分析することが望まれる。機能を調査・分析し、検証のポイントを明確化した上で、実際の検証作業に移ることが望ましい。また、アーキテクチャの観点では、センサ・監視装置がどのような OS によって動作しているかも重要となる。汎用 OS を使用している場合と独自 OS を使用している場合で、検出すべき脆弱性の特性が大きく異なるため、検証実施前に確認することが必要である。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。また、ファジングツールを用いたスキャンで検証を実施した場合、検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望まれる。

ファームウェア解析に当たって、ファームウェアの抽出可否の確認にはハードウェアを手動で解析する必

要があり、費用が高くなる傾向がある。そのため、検証依頼者からファームウェアファイルを別途受領し、ファームウェアの抽出可否の確認はせず、ファームウェアファイルの解析のみを行う場合もある。なお、検証を実施するために、検証対象機器を分解することがある。検証対象機器を分解した後に、原状復帰することが困難な可能性がある場合は、事前に検証依頼者へ伝え、検証依頼者からの合意を得た後に検証を実施することが望まれる。

バイナリ解析に当たって、抽出・受領したファームウェア等のバイナリファイルについて、静的解析ツールを用い、脆弱性の可能性のある箇所を特定するだけでなく、特定した箇所の脆弱性の再現性（悪用可能性）を確認することが必要である。脆弱性の再現性（悪用可能性）を確認するためにも、適宜、リバースエンジニアリングツールを用い、バイナリファイルをデコンパイルし、コードを確認することが必要である。

ソースコード解析に当たって、自動化ツールを活用して検出した脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。ただし、検出された脆弱性が多く、すべてを網羅的に確認することが現実的でない場合、脆弱性の深刻度や検証の工数等を鑑み、再現性の確認を行う脆弱性を決定することが必要である。

センサ・監視装置は取得したデータをクラウドサーバと接続し、保存しているケースが考えられる。本手引きでは、クラウドサーバを対象とする検証は対象外としているが、クラウドサーバにおいては、悪意ある第三者によってデータの改ざんや搾取、機能を停止される等の脅威も想定される。もし、検証依頼者の依頼を踏まえクラウドサーバに対する検証を行う場合、通信インタフェースやクラウドサーバの管理コンソールに対する検証を実施することが望ましい。なお、稼働中のクラウドサーバに対して検証を行う場合、他のサービスに影響を与えないか等を事前に検証依頼者と協議し、双方の合意の下で検証を実施することが必要である。

11 産業用コントローラに関するセキュリティ検証プラクティス

11.1 機器の概要・想定脅威

産業用コントローラとは、産業用機器の稼働及び制御する機器である。一例として、産業用コントローラの想定利用環境を図 11-1 に示す。

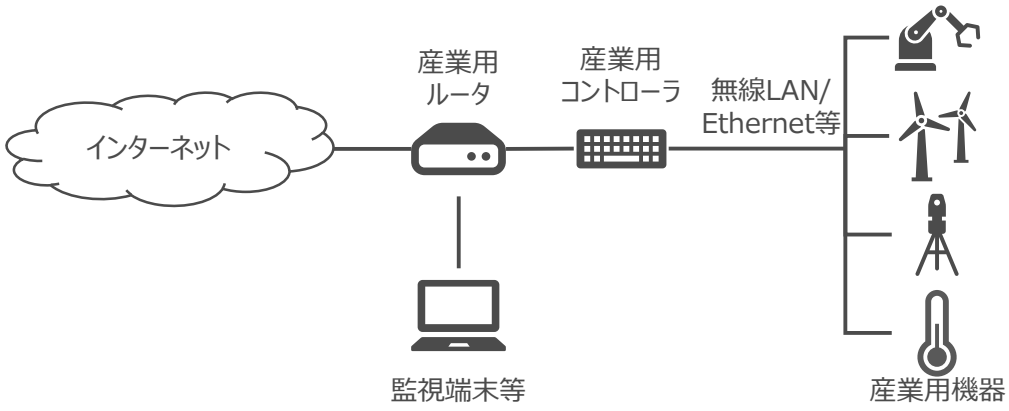

図 11-1 産業用コントローラの想定利用環境

産業用コントローラの想定脅威として、悪意ある第三者によってネットワーク機能や制御機能が阻害・停止されることにある。サイバー攻撃によりこれらの機能を停止するためには、脆弱性を悪用した内部侵入・権限昇格や、予期しないパケットやコマンドをゲートウェイ・ルータに送信することが考えられる。侵入方法としては、開発用機能をバックドアとして悪用するほか、外部通信インタフェースの脆弱性を悪用する方法等が想定されるため、特に通信インタフェースに関する脅威に対する検証が求められる。また、機器設定用の Web コンソールが用意されている場合が多く、Web コンソールに内在しうる脆弱性として、クロスサイトスクリプティングや OS コマンドインジェクション等の入力検証の不備が考えられる。これらの脆弱性を洗い出す目的で、Web コンソールに対して、既知の攻撃手法に対する影響がないかを確認することが望ましい。

11.2 想定される検証環境

産業用コントローラの検証に当たって想定される検証環境は図 11-2 に示すとおりであり、検証事業者内に検証専用の LAN を構築して検証することが望まれる。なお、産業用コントローラが接続する産業用機器の用意が難しい場合、産業用機器の通信プロトコルを処理することができるシミュレータを用意して検証することが望まれる。

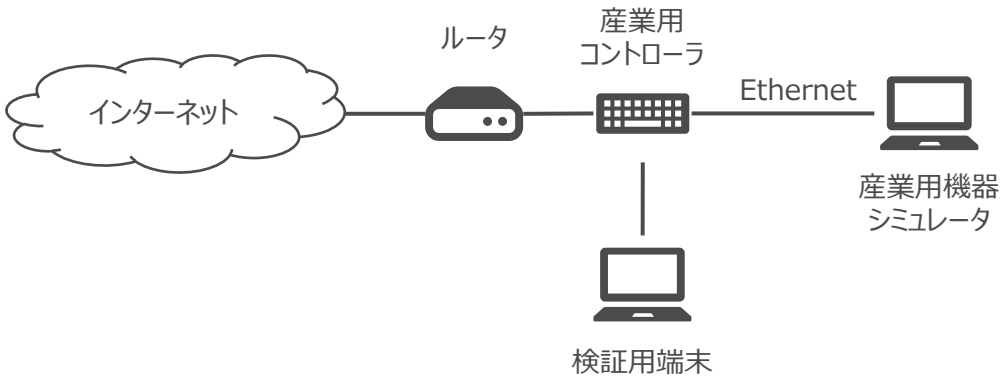

図 11-2 産業用コントローラに対する想定検証環境

11.3 実証において適用された検証手法

実証における産業用コントローラの検証では、以下に示す検証手法が適用され、検証対象機器における脆弱性の有無や脅威に対する対策の妥当性が確認された。

（1） 設計文書レビュー

検証対象機器の設計書、画面・インタフェース仕様書、マニュアル等の文書を確認し、脆弱性につながりうるセキュリティ対策上の懸念事項が含まれていないかを確認する。主な確認観点は、古いバージョンのサービスやアプリケーションによる既知脆弱性の可能性、不適切な認証情報の管理、暗号化方式の妥当性、不適切なログ設定やバックアップ設定等が挙げられる。

（2） ネットワークスキャン

検証対象機器に対してネットワークスキャンを行い、公開されているサービス及びバージョンをリストアップして、不要なサービスやポートが開放していないかを確認する。ネットワークスキャンの対象範囲として、最低限 Ethernet を対象とし、TCP 及び UDP のすべてのポート（0～65535）をスキャン対象とすることが望まれる。

検証の詳細手順：手引き別冊 1 第 4.5 節参照

（3） 既知脆弱性の診断

ネットワークスキャンの結果明らかとなったサービスやポートに既知の脆弱性が存在しないかを確認する。また、Web コンソールに対して自動化ツールを用いた脆弱性スキャンを行い、既知の攻撃手法に対する影響がないかを確認する。自動化ツールを用いた脆弱性スキャンでは、影響の有無に関わらず大量の脆弱性が出力される場合がある。検出された脆弱性については、実際にその脆弱性を悪用可能であるか、

再現性の確認を行うことが望ましい。ただし、出力されたすべての脆弱性について再現性を確認することが困難な場合、OS コマンドインジェクション、SQL インジェクション、ディレクトリトラバーサル、クロスサイトスクリプティング、XML 外部エンティティ参照（XXE）、クロスサイトリクエストフォージェリ（CSRF）等、影響の大きな脆弱性に絞って重点的に確認する。

検証の詳細手順：手引き別冊 1 第 4.6 節参照

（4） ファジング

検証対象機器の通信インタフェースや Web コンソールに対して不正なデータを送信し、アプリケーションや OS の応答有無を確認することで、未知の脆弱性の有無や機能停止に至ることがないかを確認する。機能停止に対する防御性能を確認するためには、フラッド攻撃をはじめとする機器のリソースを消費される攻撃手法を試行し、サービス不能状態に陥ることがないかを確認することが有効である。Web コンソールに対しては、ファジングツールを用いたスキャンで検証を自動化することが有効である。ただし、ファジングツールにより検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望ましい。

検証の詳細手順：手引き別冊 1 第 4.7 節参照

（5） ネットワークキャプチャ

検証対象機器の通信内容を取得し、平文の通信、暗号強度の低い通信、想定外の通信先への通信の有無を確認する。検証対象範囲について、最低限 Ethernet の通信に関するキャプチャを行うことが望まれる。また、ネットワークキャプチャの対象時間について、最低限 60 分程度の通信をキャプチャし、通信先・通信内容を確認することが望まれる。

検証の詳細手順：手引き別冊 1 第 4.8 節参照

（6） ファームウェア解析

検証対象機器のインタフェースを介してファームウェアへのアクセスが可能となる場合があるため、ファームウェアの抽出可否を確認する。仮にファームウェアの抽出が可能であった場合に、当該ファームウェアファイルを解析し、ファイルシステムの取り出し可否を確認する。ファイルシステムの取り出しが可能であった場合に、ファイルの内容を解析し、機微な情報の取得が可能であるか否かを確認する。

検証の詳細手順：手引き別冊 1 第 4.3 節参照

（7）　バイナリ解析

抽出したファームウェア等のバイナリファイルに対して、脆弱性の可能性のある箇所を確認する。脆弱性の可能性のある箇所が存在した場合、該当箇所で必ずしも脆弱性が発生するわけではない。そのため、リバースエンジニアリングツールを用いてバイナリファイルをデコンパイルし、ソースコードを確認した上で、当該脆弱性の悪用可能性を確認することが望ましい。

検証の詳細手順：手引き別冊 1 第 4.4 節参照

11.4 実証において検出された代表的な脆弱性

実証事業で検証対象とした産業用コントローラにおいて検出された代表的な脆弱性は以下のとおりである。

表 11-1 実証において検出された代表的な脆弱性

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
1	古いバージョンの Apache HTTP Server における認証回避やオーバーフローの脆弱性	悪用されることで、不正アクセスされ、サービス拒否や任意のコードを実行される可能性がある。この結果、重要情報の漏えい、産業用コントローラの動作が停止し、当該産業用コントローラを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索する。ことで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
2	古いバージョンの Apache httpd における認証回避やバッファエラーの脆弱性	悪用されることで、不正アクセスされ、サービス拒否や任意のコードを実行される可能性がある。この結果、重要情報の漏えい、産業用コントローラの動作が停止し、当該産業用コントローラを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
3	古いバージョンの OpenSSL におけるオーバーフローや任意コード実行の脆弱性	悪用されることで、予期しない動作の実行を受け、サービス拒否や任意のコードを実行される可能性がある。この結果、産業用コントローラの動作が停止し、当該産業用コントローラを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。

項番	脆弱性の概要	想定される影響	脆弱性の検出に至った検証プロセス
4	古いバージョンの glibc におけるバッファオーバーフローの脆弱性	悪用されることで、細工された不正なリクエストを受け、サービス拒否や任意のコードを実行される可能性がある。この結果、産業用コントローラの動作が停止し、当該産業用コントローラを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。
5	古いバージョンの PHP における任意コード実行、OS コマンドインジェクション、オーバーフロー等の脆弱性	悪用されることで、サービス拒否や任意のコードを実行される可能性がある。この結果、産業用コントローラの動作が停止し、当該産業用コントローラを導入している企業の業務や導入しているシステムの運用が停止するおそれがある。	・ Nmap を用いたネットワークスキャンを行い、検証対象機器で公開されているサービス及びバージョンをリストアップする。 ・ リストアップしたサービス及びバージョンの情報に基づき、既知の脆弱性が公開されていないかを NIST NVD を用いて確認する。 ・ 既知の脆弱性が公開されていたため、当該脆弱性の再現性（悪用可能性）を確認する。Exploit DB、GitHub、Metasploit 等で攻撃実証コードを検索することで、既に公開されている実証コードを用いて脆弱性の悪用可能性を確認する。

11.5 想定される推奨事項

表 11-1 で示した、実証事業で検証対象とした産業用コントローラにおいて検出された深刻度の高い脆弱性について、以下のような対策が推奨される。

表 11-2 検出された深刻度の高い脆弱性に対して推奨される対策事項

項番	脆弱性の概要	推奨される対策事項
1	古いバージョンの Apache HTTP Server における認証回避やオーバーフローの脆弱性	・ 最新の Apache HTTP Server にバージョンアップする。
2	古いバージョンの Apache httpd における認証回避やバッファエラーの脆弱性	・ 最新の Apache httpd にバージョンアップする。
3	古いバージョンの OpenSSL におけるオーバーフローや任意コード実行の脆弱性	・ 最新の OpenSSL にバージョンアップする。
4	古いバージョンの glibc におけるバッファオーバーフローの脆弱性	・ 最新の glibc にバージョンアップする。
5	古いバージョンの PHP における任意コード実行、OS コマンドインジェクション、オーバーフロー等の脆弱性	・ 最新の PHP にバージョンアップする。

11.6 検証に当たっての留意事項

産業用コントローラは、一般的には単体で利用できるものではなく、各種の機能を備えた I/O ユニットや専用のアプリケーションの基で動作が可能となる。そのため、どのような動作環境を用意し、どのような脅威を想定するかをあらかじめ設定・分析しておくことが重要である。特に、産業用機器については、特定組織内で外部のネットワークとは遮断された環境に設置され、専用の管理者が操作することを想定している場合もある。検証対象となる機器の仕様や想定動作環境については、検証依頼者との調整を十分に行う必要がある。

既知脆弱性の診断に当たって、ネットワークスキャンで明らかになったサービス・バージョン情報を踏まえて、NIST NVD 等のデータベースで脆弱性に関する情報を確認するだけでなく、その再現性（悪用可能性）を確認することが必要である。すべての脆弱性を網羅的に検証することは現実的ではないため、脆弱性の深刻度や検証の工数等を鑑み、再現性の検証を行う脆弱性を決定することが必要である。

ファジングに当たって、検証対象機器のどのプログラムに対してファジングを行うかをまず明確にすることが重要である。網羅的なファジングを行うことは現実的ではないため、検証対象機器の機能や特性を踏まえつつ、ファズデータを取捨選択することが重要である。また、ファジングツールを用いたスキャンで検証を実施した場合、検出された脆弱性については、実際にその脆弱性を悪用可能であるか、再現性の確認を行うことが望まれる。

ファームウェア解析に当たって、ファームウェアの抽出可否の確認にはハードウェアを手動で解析する必要があり、費用が高くなる傾向がある。そのため、検証依頼者からファームウェアファイルを別途受領し、ファームウェアの抽出可否の確認はせず、ファームウェアファイルの解析のみを行う場合もある。なお、検証を実施するために、検証対象機器を分解することがある。検証対象機器を分解した後に、原状復帰することが困難な可能性がある場合は、事前に検証依頼者へ伝え、検証依頼者からの合意を得た後に検証を

実施することが望まれる。

バイナリ解析に当たって、抽出・受領したファームウェア等のバイナリファイルについて、静的解析ツールを用い、脆弱性の可能性のある箇所を特定するだけでなく、特定した箇所の脆弱性の再現性（悪用可能性）を確認することが必要である。脆弱性の再現性（悪用可能性）を確認するためにも、適宜、リバースエンジニアリングツールを用い、バイナリファイルをデコンパイルし、コードを確認することが必要である。

付録 1 用語集

- **Binwalk**
  ファームウェア解析ツール。ファームウェアを展開し、ファームウェアに含まれたファイルシステムを確認することが可能。

- **Burp Suite**
  Web アプリケーションセキュリティ検証ツール。Web ブラウザと Web サーバ間の通信内容を確認することが可能。また、Web ブラウザからのリクエスト通信をキャプチャし、内容を変更して Web ブラウザへ送信することが可能。

- **CVSS（Common Vulnerability Scoring System）**
  脆弱性の深刻度を同一の基準の下で定量的に比較できる評価方法であり、0.0～10.0 の間でスコアが定まる。FIRST（Forum of Incident Response and Security Teams)が管理。

- **CWE（Common Weakness Enumeration）**
  Common Weakness Enumeration の略。ソフトウェアにおけるセキュリティ上の弱点（脆弱性）の種類を識別するための共通の基準。米国非営利団体 MITRE を中心として仕様策定。

- **DREAD**
  Damage、Reproducibility、Exploitability、Affected users、Discoverability の五つの観点の頭文字から構成される用語で、これら五つの観点に基づきリスクのスコアリングを行う手法。

- **Exploit DB**
  ソフトウェアやアプリケーションに対してサイバー攻撃を実証するための攻撃コードを管理するデータベース。米国企業の Offensive Security が管理。

- **JTAG (Joint Test Action Group)**
  IEEE1149.1 で標準化されているポートの通称。IC チップとその周辺の集積回路を含むチップセットとの相互通信や IC チップ自体の検査、回路動作に対する監視及び書き換えを行うこと等が可能。

- **IoT（Internet of Things）**
  既存又は開発中の相互運用可能な情報通信技術により、物理的又は仮想的なモノをネットワーク接続した、高度なサービスを実現するグローバルインフラ。[IoT セキュリティガイドライン ver 1.0]

- **IoT 機器（IoT Device）**
  IoT を構成する、ネットワークに接続される機器。

- **JVN（Japan Vulnerability Notes）**
  Japan Vulnerability Notes の略。脆弱性情報や対策情報を管理するポータルサイト。JPCERT/CC と IPA が共同で運営。

- **JWT（JSON Web Token）**
  JSON Web Token の略。JSON 形式で表現された認証情報を URL 文字列へ変換し、安全に送受信できるよう、符号化やデジタル署名等の仕組みを規定した標準規格。

- **Metasploit**
  サイバー攻撃を実証するためのフレームワーク（セキュリティ検証ツール）。サイバー攻撃のためのコード作成や実証を行うことが可能。

- **MQTT（Message Queuing Telemetry Transport）**
  Message Queuing Telemetry Transport の略。多数の通信主体の間で短いメッセージを頻繁に送受信する用途に向いた通信規約。

- **Nmap**
  ネットワーク調査ツール。調査対象の機器にアクセスを試行し、開放されたポートや動作しているシステムを調査することが可能。

- **NVD（National Vulnerability Database）**
  National Vulnerability Database の略。NIST が運営する脆弱性情報データベースのこと。

- **OS コマンドインジェクション（OS Command Injection）**
  OS コマンドを不正に埋め込み、標的対象を不正に操作するサイバー攻撃。

- **OWASP（Open Web Application Security Project）**
  Web をはじめとするソフトウェアのセキュリティ関する情報共有と普及啓発を目的とした、オープンソース・ソフトウェアコミュニティ。

- **OWASP ZAP**
  OWASP が提供する Web アプリケーションセキュリティ検証ツール。検証対象の URL を入力することで、静的脆弱性スキャン・動的脆弱性スキャンが可能。

- **SPI (Serial Peripheral Interface)**
  回路基板上の各 IC チップを接続するために使用されるインタフェース。

- **SQL インジェクション（SQL Injection）**
  ユーザが入力する検索文字列等の外部から指定するパラメータに SQL 文を混入させ、標的対象のデータベースを不正に操作するサイバー攻撃。

- **UART (Universal Asynchronous Receiver/Transmitter)**
  デバッグ等を目的として、外部端末から回路基板にアクセスするために使用されるシリアル信号とパラレル信号の変換を行う集積回路。

- **Wireshark**

ネットワーク通信解析ツール。ネットワーク上の通信をキャプチャし、キャプチャした通信を解析することで、通信プロトコル、送信元、宛先等の情報を確認することが可能。

- **XML 外部エンティティ参照（XML External Entity）**
  XML の外部ファイル取り込みの機能を悪用し、サーバ内のファイル等を不正に取り込み、重要情報の漏えい等を引き起こすサイバー攻撃。

- **脅威（Threat）**
  システム又は組織に損害を与える可能性がある、望ましくないインシデントの潜在的な原因。[JIS Q 27000:2014]

- **脅威分析（Threat Analysis）**
  機器やソフトウェア、システム等に対する脅威を抽出し、その影響を評価すること。主に、製品の要件定義、設計フェーズにて行われる。

- **クロスサイトスクリプティング（Cross-site Scripting）**
  Web サイトに存在する脆弱性を悪用し、ユーザが Web サイトにアクセスすることで、攻撃者が用意した悪意のあるスクリプトをユーザに実行させるサイバー攻撃。

- **クロスサイトリクエストフォージェリ（CSRF : Cross-site Request Forgeries）**
  Web サイトのセッション管理に関する脆弱性を悪用し、ユーザが Web サイトにアクセスすることで、ユーザの意図しない Web アプリケーション上の処理を実行させるサイバー攻撃。

- **サイバー攻撃（Cyber Attack）**
  資産の破壊、暴露、改ざん、無効化、盗用、又は認可されていないアクセス若しくは使用の試み。[JIS Q 27000:2014]

- **サイバーセキュリティ（Cybersecurity）**
  電子データの漏えい・改ざん等や、期待されていた機器、IT システム、制御システム等の機能が果たされないといった不具合が生じないようにすること。

- **サプライチェーン（Supply Chain）**
  複数の開発者間でリンクされたリソース・プロセスで、製品とサービスについて、調達にはじまり設計・開発・製造・加工・販売及び購入者への配送に至る一連の流れ。[ISO 28001:2007, NIST SP 800-53 Rev.4]

- **シグネチャ（Signature）**
  通信パケットに含まれる、攻撃に関係する認識可能で特徴的なパターン。ウイルス中のバイナリ文字列や、システムへの不正アクセスを得るために使用する特定のキーストロークなど。[NIST SP 800-61 Rev.1]

- **脆弱性（Vulnerability）**
  一つ以上の脅威によって付け込まれる可能性のある、資産又は管理策の弱点。[JIS Q 27000:2014]

- **脆弱性検証（Vulnerability Validation）**
  脆弱性の存在を確認するアクティブなセキュリティ検証手法。[NIST SP 800-115]
  脆弱性を洗い出すことを目的とする。

- **セキュリティ検証（Security Validation）**
  機器、システム、組織における脅威に対するセキュリティ対策の妥当性や脆弱性の有無を確認する手法。本手引きでは、特に機器に対するセキュリティ検証について記載している。

- **ディレクトリトラバーサル（Directory Traversal）**
  ファイル名を扱うプログラムに対して特殊な文字列を含むファイル名を送信し、通常はアクセスできないファイルやディレクトリの内容を窃取するサイバー攻撃。

- **認証（Authentication）**
  エンティティの主張する特性が正しいという保証の提供。[JIS Q 27000:2014]

- **認可（Authorization）**
  アクセス権限に基づいたアクセス機能の提供を含む権限の付与 [ISO 7498-2:1989]

- **バックドア（Backdoor）**
  機器に設けられた、正規のログイン方法ではない非公表のアクセス方法。潜在的なセキュリティリスクとなりうる。[NIST SP 800-82 Rev.2]

- **ファジング（Fuzzing）**
  検証対象の機器やソフトウェアに脆弱性を引き起こしうるデータ（ファズデータ）を送り込み、その挙動を確認することで脆弱性を検出する手法。

- **プロトコル（Protocol）**
  複数の主体が滞りなく信号やデータ、情報を相互に伝送できるよう、あらかじめ決められた約束事や手順の集合のこと。

- **ペネトレーションテスト（Penetration Test）**
  組織が有するすべてのシステムや、指定されたシステム全体を対象とし、明確な意図を持った攻撃者によって、その目的が達成されうるかを確認するセキュリティ検証手法。

- **マルウェア（Malware）**
  許可されていないプロセスの実施を試みることによって、情報システムの機密性・完全性・可用性に悪影響をもたらすソフトウェア又はファームウェア。[NIST SP 800-53 Rev.4]

セキュリティ上の被害を及ぼすウイルス、スパイウエア、ボット等の悪意を持ったプログラムを指す総称。

- **リスク（Risk）**
  目的に対する不確かさの影響。[JIS Q 27000:2014]

付録 2 実証において検出された代表的な脆弱性

本付録では、令和 4 年度に実施した中小企業等が開発する 155 の IoT 製品に対するセキュリティ検証の実証において検出された脆弱性のうち、深刻度が高い代表的な脆弱性について、脆弱性が検出された機器区分、当該脆弱性の概要と検証に当たって活用された情報（設計書、仕様書等の文書、ファームウェア、ソースコード、プロトタイプ、機器本体等）を示す。

機器区分	検出された脆弱性	検証に当たって活用された情報							
	脆弱性の概要	1.設計書	2.仕様書	3.マニュアル	4.その他関連文書	5.ファームウェア	6.ソースコード	7.プロトタイプ	8.機器本体
UTM	古いバージョンの OpenSSH における権限昇格の脆弱性（CVE-2021-41617）		○	○					○
UTM	Web 管理画面における OS コマンドインジェクションの脆弱性		○	○					○
UTM	非暗号化通信によるファームウェアの更新		○	○					○
ゲートウェイ・ルータ	古いバージョンの Squid における不正アクセスやリモートコード実行の脆弱性（CVE-2019-12519、CVE-2019-12523、CVE-2019-12524、CVE-2019-12525、CVE-2019-12526、CVE-2020-11945）		○	○					○
ゲートウェイ・ルータ	古いバージョンの dnsmasq におけるバッファオーバーフローの脆弱性（CVE-2017-14491、CVE-2017-14492、CVE-2017-14493）			○		○			○
ゲートウェイ・ルータ	古いバージョンの lighttpd における SQL インジェクションの脆弱性（CVE-2014-2323）			○		○			○

機器区分	検出された脆弱性	検証に当たって活用された情報							
	脆弱性の概要	1.設計書	2.仕様書	3.マニュアル	4.その他関連文書	5.ファームウェア	6.ソースコード	7.プロトタイプ	8.機器本体
ゲートウェイ・ルータ	バッファオーバーフローの脆弱性			○		○			○
ゲートウェイ・ルータ	古いバージョンの OS における不正アクセスやリモートコード実行の脆弱性			○		○			○
ゲートウェイ・ルータ	古いバージョンの Samba におけるリモートコード実行の脆弱性（CVE-2017-7494）			○		○			○
ゲートウェイ・ルータ	アップデートサーバ及びアップデートファイルの署名検証不備			○		○			○
ネットワークスイッチ	古いバージョンの Dropbear における情報の漏えいや改ざんの脆弱性（CVE-2017-9078、CVE-2019-12953、CVE-2020-36254）			○					○
ネットワークスイッチ	Web 管理画面に対する非暗号化状態での通信			○					○
ネットワークスイッチ	Web 管理画面におけるクロスサイトスクリプティングの脆弱性			○					○
ネットワークスイッチ	Web 管理画面における画像アップロード機能において任意のファイルをアップロード可能			○					○
モバイル端末	初期インストールされた Web アプリケーションにおけるディレクトリトラバーサルの脆弱性		○	○					○

機器区分	検出された脆弱性	検証に当たって活用された情報							
	脆弱性の概要	1.設計書	2.仕様書	3.マニュアル	4.その他関連文書	5.ファームウェア	6.ソースコード	7.プロトタイプ	8.機器本体
モバイル端末	タッチパネル操作によって機密情報の閲覧が可能		○	○					○
モバイル端末	誰もが匿名で利用可能なサーバが稼働		○	○					○
モバイル端末	初期インストールされた Web アプリケーションにおいて権限を改ざんした操作が可能		○	○					○
スマートロック	Web アプリケーションにおいて他ユーザの情報変更が可能		○	○			○		○
スマートロック	Web アプリケーションにおけるクロスサイトスクリプティングの脆弱性		○	○			○		○
スマートロック	古いバージョンの systemd におけるリモートコード実行の脆弱性（CVE-2022-2526）		○	○					○
スマートロック	Web アプリケーションにおけるユーザのパスワード攻撃の脆弱性		○	○					○
スマートロック	MQTT（Message Queuing Telemetry Transport）通信における認証制御の不備		○	○					○
スマート家電	Web アプリケーションにおいて他ユーザの情報変更が可能		○	○			○		○
スマート家電	adb（android debug bridge）を利用したスマート家電への接続・任意操作可能		○	○			○		○
スマート家電	Web アプリケーションにおける OS コマンドインジェクションの脆弱性		○	○			○		○
スマート家電	Web アプリケーションにおけるファイルアップロード機能において任意のファイルをアップロード可能		○	○			○		○
スマート家電	Web アプリケーションにおけるバッファオーバーフローの脆弱性		○	○			○		○

機器区分	検出された脆弱性	検証に当たって活用された情報							
	脆弱性の概要	1.設計書	2.仕様書	3.マニュアル	4.その他関連文書	5.ファームウェア	6.ソースコード	7.プロトタイプ	8.機器本体
ドローン	ドローン本体のネットワーク設定機能における OS コマンドインジェクションの脆弱性	○	○	○		○	○		
ドローン	Web アプリケーションにおけるファイルアップロード機能において任意のコードを実行可能	○	○	○		○	○		
ドローン	Web アプリケーションにおける OS コマンドインジェクションの脆弱性	○	○	○		○	○		
ドローン	公開フォルダ上に機密情報が設置	○	○	○		○	○		
ネットワークカメラ	Web アプリケーションにおける OS コマンドインジェクションの脆弱性		○	○		○			○
ネットワークカメラ	Web アプリケーションおける認証制御の不備		○	○		○			○
ネットワークカメラ	Web アプリケーションにおけるクロスサイトリクエストフォージェリの脆弱性		○	○		○			○
ネットワークカメラ	不正な SD カードを挿入することによる任意コード実行の脆弱性		○	○		○			○
ネットワークカメラ	Web アプリケーションへのログインパスワードを不正取得可能		○	○		○			○
ネットワークカメラ	アップデート用ファームウェアファイルから重要情報取得可能		○	○		○			○

機器区分	検出された脆弱性	検証に当たって活用された情報							
	脆弱性の概要	1.設計書	2.仕様書	3.マニュアル	4.その他関連文書	5.ファームウェア	6.ソースコード	7.プロトタイプ	8.機器本体
センサ・監視装置	古いバージョンの dnsmasq におけるバッファオーバーフローの脆弱性（CVE-2017-14491、CVE-2017-14492、CVE-2017-14493）			○		○			○
センサ・監視装置	Web アプリケーションにおける OS コマンドインジェクションの脆弱性			○					○
センサ・監視装置	DLL ファイルにおける既知のディレクトリトラバーサルの脆弱性								○
センサ・監視装置	管理者権限ユーザの作成における認証回避の脆弱性				○	○			○
センサ・監視装置	Web アプリケーションにおける任意コード実行の脆弱性				○	○			○
センサ・監視装置	OS におけるアカウント認証情報を推測可能				○	○			○
センサ・監視装置	古いバージョンの Grails が使用している外部ライブラリにおける XML 外部エンティティ参照（XXE）の脆弱性				○	○			○
センサ・監視装置	Web アプリケーションにおけるファイルアップロード機能において任意のファイルをアップロード可能				○	○			○
センサ・監視装置	開放されたポートを経由した不正接続				○	○			○

機器区分	検出された脆弱性	検証に当たって活用された情報							
	脆弱性の概要	1.設計書	2.仕様書	3.マニュアル	4.その他関連文書	5.ファームウェア	6.ソースコード	7.プロトタイプ	8.機器本体
産業用コントローラ	古いバージョンの Apache HTTP Server における認証回避やオーバーフローの脆弱性（CVE-2021-26691、CVE-2021-39275、CVE-2021-44790、CVE-2022-22720、CVE-2022-22721、CVE-2022-28615、CVE-2022-31813）			○		○			○
産業用コントローラ	古いバージョンの Apache httpd における認証回避やバッファエラーの脆弱性（CVE-2017-3167、CVE-2017-7679）			○		○			○
産業用コントローラ	古いバージョンの OpenSSL におけるオーバーフローや任意コード実行の脆弱性（CVE-2016-2108、CVE-2016-2177、CVE-2016-6303）			○		○			○
産業用コントローラ	古いバージョンの glibc におけるバッファオーバーフローの脆弱性（CVE-2015-0235）			○		○			○

機器区分	検出された脆弱性	検証に当たって活用された情報							
	脆弱性の概要	1.設計書	2.仕様書	3.マニュアル	4.その他関連文書	5.ファームウェア	6.ソースコード	7.プロトタイプ	8.機器本体
産業用コントローラ	古いバージョンの PHP における任意コード実行、OS コマンドインジェクション、オーバーフロー等の脆弱性（CVE-2015-4116、CVE-2015-4599、CVE-2015-4600、CVE-2015-4601、CVE-2015-4602、CVE-2015-4603、CVE-2015-4604、CVE-2015-4643、CVE-2015-5589、CVE-2015-6834、CVE-2015-6835、CVE-2015-8876、CVE-2016-3141、CVE-2016-4071、CVE-2016-4072、CVE-2016-4073、CVE-2016-4537、CVE-2016-4538、CVE-2016-4539、CVE-2016-4540、CVE-2016-4541、CVE-2016-4542、CVE-2016-4543、CVE-2016-4544、CVE-2016-5114、CVE-2016-5768、CVE-2016-5769、CVE-2016-5770、CVE-2016-5771、CVE-2016-5772、CVE-2016-5773、CVE-2016-6290、CVE-2016-6291、CVE-2016-6294、CVE-2016-6295、CVE-2016-6296、CVE-2016-7126、CVE-2016-7127、CVE-2016-7411、CVE-2016-7413、CVE-2016-7414、CVE-2016-7417、CVE-2016-7480、CVE-2016-7568、CVE-2016-8670 等）			○		○			○
CPU ボード	プロセッサチップにおける既知脆弱性（CVE-2021-0146）			○					○
産業用ロボット	USB 接続による任意ファイル起動可能な脆弱性	○	○	○					○

機器区分	検出された脆弱性	検証に当たって活用された情報							
	脆弱性の概要	1.設計書	2.仕様書	3.マニュアル	4.その他関連文書	5.ファームウェア	6.ソースコード	7.プロトタイプ	8.機器本体
産業用ロボット	古いバージョンの dnsmasq におけるバッファオーバーフローの脆弱性（CVE-2017-14491、CVE-2017-14492、CVE-2017-14493）	○	○	○					○
サーバ装置	OpenSSH における権限窃取の脆弱性（CVE-2016-1908）			○		○			○
サーバ装置	dnsmasq におけるバッファオーバーフローの脆弱性（CVE-2017-14491、CVE-2017-14492、CVE-2017-14493）			○		○			○
複合機	認証情報における脆弱なパスワードの設定			○		○			○