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 2　機器メーカに向けた脅威分析及びセキュリティ検証の解説書

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

1.1 IoT セキュリティの現状と課題 1

1.2 本別冊の目的 1

1.3 本別冊で対象とする機器 1

1.4 対象者 2

1.5 本別冊の活用方法 2

1.6 本別冊の構成 3

2 IoT 機器等開発における検証の位置づけ 5

2.1 機器開発プロセスにおける検証フェーズの位置づけと目的 5

2.2 検証の実施効果 5

3 機器メーカにおける脅威分析の実施 6

3.1 守るべき資産の検討 6

3.2 攻撃ポイントの分析 8

3.3 想定される脅威の分析 10

3.4 セキュリティ要求事項の検討 16

4 検証依頼にあたって機器メーカが知るべき検証手法 24

4.1 機器メーカが知るべき代表的な検証手法 24

4.2 ハードウェアハッキング・ファームウェア解析 26

4.3 バイナリ解析 32

4.4 ネットワークスキャン 35

4.5 既知脆弱性の診断 36

4.6 ファジング 42

4.7 ネットワークキャプチャ 44

4.8 検証依頼時に用意することが望まれる情報 47

5 検証結果を踏まえた対応の検討 50

5.1 検証報告に対する機器メーカの対応 50

5.2 検証結果に基づくリスク評価と対応方針の検討 53

5.3 ハードウェアハッキング・ファームウェア解析の結果を踏まえての対応 59

5.4 バイナリ解析の結果を踏まえての対応 59

5.5 ネットワークスキャンの結果を踏まえての対応 60

5.6 既知脆弱性の診断の結果を踏まえての対応 60

5.7 ファジングの結果を踏まえての対応 61

5.8 ネットワークキャプチャの結果を踏まえての対応 62

5.9 製品リリースにあたり機器メーカとして検討しておくべき事項 62

6 付録 67
6.1 国内外のセキュリティガイドラインと本別冊との対応 67
6.2 検証依頼時に用意することが望まれる情報の逆引き表 70
6.3 用語集 71
6.4 参考文書 75

1 背景と目的

1.1 IoT セキュリティの現状と課題

数多くの機器に ICT（情報通信）技術が組み込まれ、機器がインターネットにつながる IoT（Internet of Things）の時代が到来し、製品のさらなる高度化、自動化を実現することとなった。我が国においても、平成 28 年 1 月 22 日に閣議決定された「第 5 期科学技術基本計画」において、サイバー空間とフィジカル空間を高度に融合させることにより、多様なニーズにきめ細かに対応したモノやサービスを提供し、経済的発展と社会的課題の解決を両立する超スマート社会「Society5.0」を提唱している。「Society5.0」と、様々なつながりによって新たな付加価値を創出する「Connected Industries」では、IoT によって製品やサービスに対する新たな価値の創出が期待される一方、企業やサプライチェーンについては、より複雑化したサイバーセキュリティ上の脅威に対応していく必要がある。

実際に 2016 年には米国や欧州でマルウェア「Mirai」によるかつてない大規模な DDoS 攻撃が行われ、インターネット関連企業や、サーバプロバイダに大きな被害を与えた。同マルウェアに感染している機器は、世界に数十万台ともいわれており、その多くは特定しやすい ID やパスワードの設定や telnet サービスポートの開放といった基本的な問題に起因している。このことはこれまで想定でしかなかった IoT 機器等のネットワークアクセスによるマルウェアや不正アクセスなどの脅威が、実際にユーザやメーカの安全・安心を阻害しうることを実証し、産業界に生活機器のセキュリティ対策を促す大きな警鐘となった。

民間で利用される多くの IoT 機器等は、その運用が一般の利用者に委ねられており、明確な管理者が設定されていないことがほとんどである。そのため、製品ライフサイクルを通じて安心・安全な IoT 機器等を利用者に提供するためには、機器を開発、生産、販売するメーカにより機器のセキュリティを担保する必要がある。機器のセキュリティを担保するためには、機器に想定される想定されるリスクの分析や、事前の対策を含んだリスク管理、そしてリスク対策の妥当性を確認するためのセキュリティ検証の実施が必要となる。

1.2 本別冊の目的

本別冊は、IoT 機器等を開発、生産、販売するメーカを対象に、セキュリティ対策の指針となる事項を示すものである。メーカが検証サービス事業者に、セキュリティ検証（以降、省略し「検証」という）を依頼するにあたっては、目的と合致した依頼を行うために、メーカ側も十分な検証の知識を有することが望ましい。また、効果的な検証を行うためには、事前の脅威分析や、開発段階において適切なセキュリティ対策が行われていることが必要となる。検証依頼者であるメーカが本別冊を活用することで、二者間において、円滑なコミュニケーションが行われ、目的に適った効率的かつ効果的な検証体制が構築されることが期待される。

1.3 本別冊で対象とする機器

本別冊の対象機、IoT 機器等をはじめとするネットワークに常時接続する機器とする。

1.4　対象者

本別冊は、IoT 機器等の開発、生産、販売を行うメーカの開発者、品質保証部門の担当者、セキュリティ担当者を主な対象とし、IoT 機器等の開発における対策のポイントを示すものである。また、副次的にはメーカ側のニーズを理解する参考資料として、検証サービス事業者も対象としている。

1.5　本別冊の活用方法

本別冊は、メーカの開発者の観点から、検証フェーズを中心に「検証依頼前に実施しておくべき内容」、「検証依頼時に知っておきたい内容」、「検証完了後に実施する内容」の三つに分類している。

「検証依頼前に実施しておくべき内容」としては、まずセキュリティ対策の指針となる脅威分析を行う。対象機器におけるシステム構成を踏まえて、機器を使用する上で必要となるセキュリティ機能の要求事項や、脅威となりえるポイントを抽出し、対策すべき事項を整理する。脅威分析については本別冊の「3 機器メーカにおける脅威分析の実施」を参照されたい。

「検証を依頼時に知っておきたい内容」としては、メーカ側が期待し、目的に適ったセキュリティ検査を依頼するために、代表的なセキュリティ検証の内容を示す。対象製品においてメーカとして優先度が高いと想定している機能に関する検証や、懸念事項や課題が残る点についての検証を把握し、検証依頼内容を決定する。セキュリティ検証については、本別冊の「4 検証依頼にあたって機器メーカが知るべき検証手法」を参照されたい。

「検証完了後に実施する内容」としては、検証により明らかになった課題や脆弱性に対するセキュリティ対策を行う。また、製品リリースにあたり、セキュリティサポートの体制や利用者への情報開示など、メーカとして対応が必要な事項についても検討を行う。セキュリティ対策については、本別冊の「5 IoT 機器等における対策のポイント」を参照されたい。

上記、本別冊の構成と検証フェーズとの対応関係については、図 1-1 に整理するとおりである。

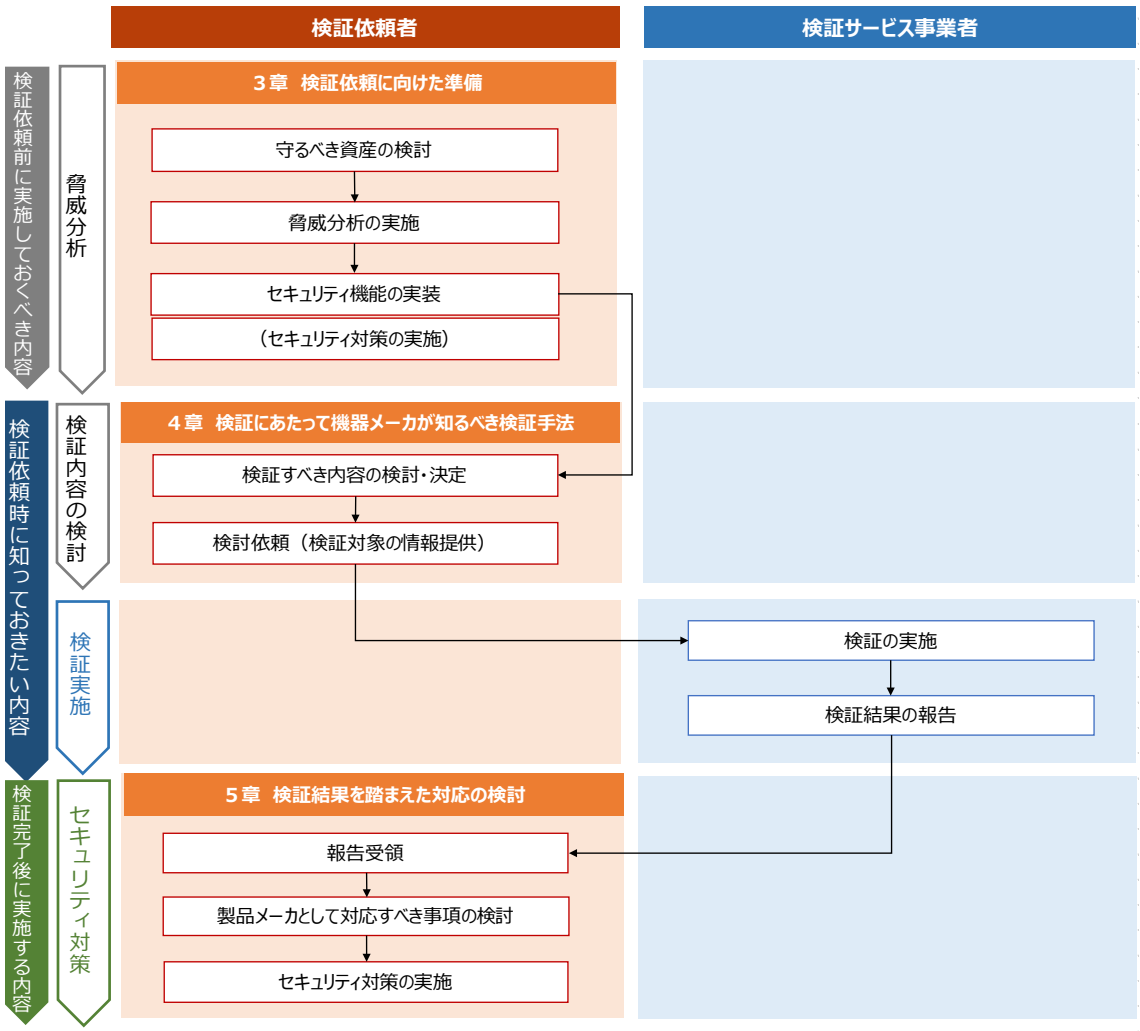

図 1-1 本別冊の構成と検証フェーズとの対応関係

1.6 本別冊の構成

第 1 章においては、本別冊の背景や目的、対象とする機器、対象者、そして活用方法を示した。

第 2 章においては、機器開発プロセスにおける検証フェーズの位置づけを示すと共にし、検証フェーズが機器開発プロセスにおいて果たすべき役割や目的、またメーカにとって期待される効果を示す。

第 3 章においては、検証依頼に向けた準備として、メーカにおけるセキュリティ対策の指針となる脅威分析の手法やプロセスについて、順を追って示す。

第 4 章においては、目的に沿った内容で検証を依頼するために、メーカが知っておくべき代表的な検証手法の概要やその効果を示す。また、検証依頼に検証サービス事業者に提示すべき情報や、検証完了後に求める報告内容についても記載する。

第 5 章においては、検証結果を踏まえ、機器メーカが実施すべきサイバー攻撃に対する対策のポイントを示す。また、セキュリティサポートの体制や利用者への情報開示など、製品リリースにあたり、メーカとして対応が必要な事項についても示す。

第 6 章においては、付録として、サンプル IoT 機器等を対象とした検証結果事例や、第 5 章で示した対策ポイントと主要なガイドラインとの対応関係を示す。また、本別冊で使用する用語の定義と参考とした文書を示す。

2 IoT 機器等開発における検証の位置づけ

2.1 機器開発プロセスにおける検証フェーズの位置づけと目的

本別冊では、機器開発プロセスにおける「検証フェーズ」を図 2-1 のように位置づける。検証フェーズは「開発・実装」フェーズ後に実施され、検証の結果、検出された課題や脆弱性（セキュリティに関するものに限らない）を製品にフィードバックし、製品品質の向上につなげることが目的となる。

またメーカとしては、検出された課題に応じて「企画・要件定義」フェーズから「開発・実装フェーズ」に至る機器開発プロセスの改善を図ることで、機器開発における PDCA サイクルを構築し、会社組織全体の向上につなげることも重要である。

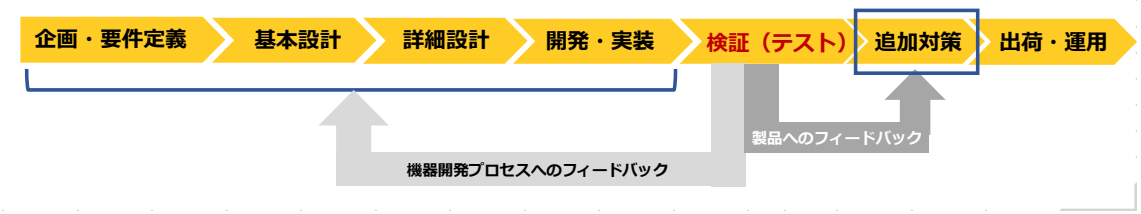

図 2-1 機器開発プロセスにおける検証フェーズの位置づけ

2.2 検証の実施効果

セキュリティリスクへの対応は、その対策範囲が多岐のフェーズに亘り、検証フェーズ単体ですべての課題や脅威を抽出することは困難であるため、「企画・要件定義」から「開発・実装」フェーズを含む広範な機器開発プロセスにおいて取り組む必要がある。機器開発プロセスの初期段階において実施する脅威分析を踏まえ、実装すべきセキュリティ機能の要件定義や、セキュリティ・バイ・デザインに沿った「基本設計」「詳細設計」「開発・実装」フェーズでの対応が、前提として求められる。こうした前提を基に検証を実施することで、前フェーズまでに行われた対策の妥当性を検証すると共に、想定外の脆弱性の検出を行うことで、製品のセキュリティレベル向上につなげることが可能となる。

また、検証の実施にあたっては、企業内の品質保証部門が実施するよりも、より高度な専門的知識や技術を有する検証サービス事業者に依頼することで、第三者的な客観性を担保し、より高い精度の検証効果を期待することができる。

3 機器メーカにおける脅威分析の実施

製品の要件定義、設計の段階で脅威分析を実施し、セキュリティ対策を行う上での指針とする。製品開発の初期フェーズにてセキュリティを考慮しておくことにより、後工程からの手戻りの削減が期待できる。本章における脅威分析手順は「IoT 開発におけるセキュリティ設計の手引き」（IPA）[1]を参考としている。

本章では、図 3-1 のような一般的な家庭用ネットワークカメラに対して脅威分析のプロセスを適用した場合の例を示す。なお、ユースケースについては、下記のように想定した。脅威分析を実施する際には、本記載例を参考に、システム構成やユースケースを事前準備として整理しておく必要がある。

- ネットワークカメラ本体は、通信インタフェースとして無線 Wi-Fi、LAN ポートを実装している。
- 撮影映像の保存先として、機器に SD カードの物理ポートを実装している。
- リモートアクセスの設定機能を実装し、スマートフォン/タブレットのスマートフォンアプリや、PC にインストールされた機器付属のアプリケーションソフトを用いて、宅外から撮影映像の視聴が可能となる。
- メーカは専用のアプリケーションサーバを有し、宅外利用時のリモートアクセスに対して通信管理を行う。

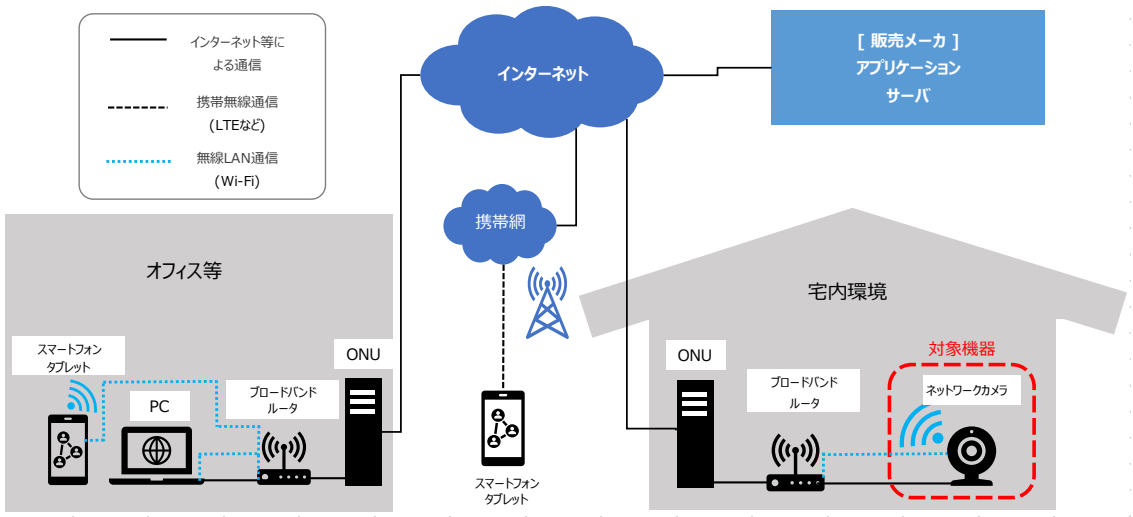

図 3-1 一般的な家庭用ネットワークカメラにおけるシステム構成例

3.1 守るべき資産の検討

脅威分析の第 1 ステップとして、守るべき資産の分析を行う。守るべき資産の該当する対象について、分類と具体例を表 3-1 に示す。個人情報や機密情報、セキュリティ情報など、情報資産として取り扱われる対象に加えて、IoT 機器等では、金融端末（ATM）や決裁端末（POS）などにおいては、モノと

[1] 独立行政法人情報処理推進機構（IPA）「IoT 開発におけるセキュリティ設計の手引き」
https://www.ipa.go.jp/security/iot/iotguide.html

して取り扱いが可能な金銭のような資産も存在する。また社会的なインフラとして重要度の高い機器、設備については、プログラムのソースコードの漏えいすることで改ざんや攻撃に悪用されるリスクがあり、これらも情報資産として含めることができる。健康、人命については、インターネット経由で直接脅威にさらされることはないが、自動制御技術の発展等により、他の資産が脅威にさらされることによって間接的な影響を受ける可能性がある。

表 3-1 守るべき資産の分類と具体例

資産の種別（大分類）	分類	具体例
物理的資産	機器本体	対象となる IoT 機器等本体
	健康、人命	機器の利用にかかわるユーザの物理的安全性
	金銭	現金取扱い端末によって扱われる現金等
データ資産	個人情報	氏名、生年月日等、個人を識別可能な情報
	機密情報	企業にとって外部への開示を予定していない情報
	金融資産に紐づくデータ	クレジットカード番号、銀行口座番号等
	設定情報	ネットワークの設定情報、アクセス権限等
	ログ情報	実際にやり取りされる通信データやクライアント、サーバの情報等
	セキュリティ情報	アクセス用の ID、パスワード情報、電子証明書、暗号鍵等
	制御信号	機器、システムへの制御に使用される情報
ソフトウェア資産	プログラムコード (ソフトウェア、ファームウェア)	対象となる IoT 機器等に含まれるソフトウェアやファームウェアなどのソースコード
サービス	機能・サービス	機器が提供する機能やサービス（可用性）

守るべき資産の選定するにあたり、設計段階で定義された機器が保持する情報や資産を対象に、インシデントが発生した場合の製品（サービス）提供や企業活動への影響を検討する。以下に具体的な影響の例を示す。

- 経済的な損失につながる
- 業務活動やサービス提供が阻害される
- 法制度に抵触する（個人情報保護法など）
- 評判、ブランドの毀損につながる
- 守るべき資産を保護するためのセキュリティ機能に影響する（機能が無効化される）

具体的に何を守るべき資産として取り扱うかについては、実際の利用シーンや上記例の影響を踏まえて、

機器メーカが個別に検討する必要がある。

一般的な家庭用ネットワークカメラに関して検討を行った場合、守るべき資産は一例として表 3-2 のように整理される。

表 3-2 一般的な家庭用ネットワークカメラにおける守るべき資産の例

資産の種別（大分類）	分類	守るべき資産	選定の理由
データ資産	個人情報	カメラ映像、録画データ	カメラ映像や録画データは、利用者のプライバシー情報が含まれるため、個人情報保護法の対象となり、守るべき資産として取り扱う必要がある。
	設定情報	ネットワークカメラの管理用機能で設定可能な設定値（リモートアクセス設定など）	設定情報は、例えば宅外からカメラ映像を確認するためのリモートアクセス設定や、IP アドレスのフィルタ設定などが含まれ、守るべき資産を保護するセキュリティ機能に影響する。
	セキュリティ情報	ユーザ認証用の ID やパスワード	セキュリティ情報は設定情報と同じく、守るべき資産を保護するセキュリティ機能に影響する。
ソフトウェア資産	プログラムコード	カメラ本体のソフトウェア、ファームウェア	プログラムコードは、自社で開発を行った場合、機器メーカにとっては、それ自体が収益の源泉であると共に、リバースエンジニアリング等により機微な情報が漏えいする可能性があるため、守るべき資産として取り扱う必要がある。
サービス	機能・サービス	ネットワークカメラが提供する機能や、サービス	機能・サービスは対象機器によるサービス提供を正常に行う上で保護される必要がある。

3.2 攻撃ポイントの分析

守るべき資産の検討後、対象機器のシステム構成図を基に脅威が想定される攻撃ポイントの分析を行う。IoT 機器等は様々な機器との接続により、脅威が想定される箇所が多岐に亘るため、システム構成図はユーザのユースシーンを踏まえた全体構成を俯瞰して検討することが望ましい。

IoT 機器等において攻撃ポイントとなりえる部位としては下記のポイントが想定される。

- **対象機器における入出力インタフェース：**

機器に対する入出力インタフェースを介した攻撃・侵入の入り口となる可能性がある。
例：Ethernet、無線 LAN（Wi-Fi）、Bluetooth、IrDA、USB、SD カード、JTAG など

- **対象機器の基板、回路上に存在する伝送経路：**
  機器の電子基板上の伝送経路に対する攻撃・侵入の入り口となる可能性がある。
  例：CPU Bus、Memory Bus など
- **システム構成上の通信経路**：
  中間者攻撃など、経路上での情報漏えい、改ざんの可能性がある。
- **上記以外の対象機器に対する物理的な接触**：
  機器の自然損壊や、破壊行為等による影響を受ける可能性がある。

入出力インタフェースが攻撃ポイントとなるケースとして、攻撃者によるインターネット経由のリモート攻撃や、マルウェアの侵入口となる可能性が想定される。ネットワークカメラにおける攻撃の事例として、Web アプリとして実装された管理画面にコマンドインジェクションに関する脆弱性があり、「telnetd」コマンドによって、telnet ポートを開放されてしまう問題が報告されている。

基板、回路上に存在する伝送経路が攻撃ポイントとなるケースとして、グリッチ攻撃のように電圧を操作し、伝送路を流れるデータを書き換える攻撃などが事例として報告されている。

システム構成上の通信経路が攻撃ポイントとなるケースとしては、Wi-Fi の暗号化方式に WEP などのセキュリティ強度が低い方式が使用されている場合、無線範囲内で解析ツールを使用することで、SSID や PSK が解析され、宅内 LAN 環境への攻撃に利用される可能性がある。

一般的な家庭用ネットワークカメラにおいて、想定される攻撃ポイントの例を図 3-2 に示す。入出力インタフェースにおける攻撃ポイントとして、宅内でブロードバンドルータとの通信に利用される無線 LAN（A）、同じく宅内で利用される LAN（B）、録画映像の保存などに使用される SD カードや USB のインタフェース（C）、そして基盤・回路上のポート（D）の 4 箇所が想定される。

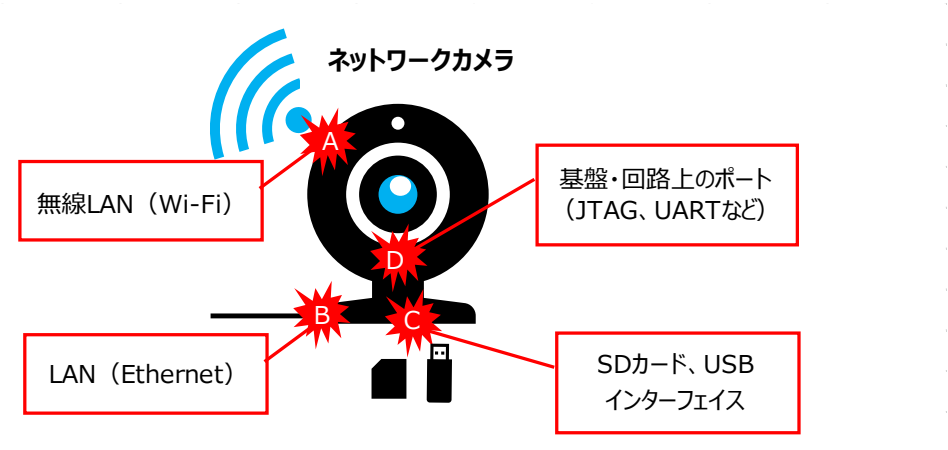

図 3-2 対象機器の入出力インタフェースにおける攻撃ポイント例

システム構成上の通信経路における攻撃ポイントを分析した場合、図 3-3 内１～６の合計６箇所が想定される。なお、スマートフォンやタブレットによる携帯無線通信については、宅外で利用される場合、通信キャリアよってセキュリティ強度の高い認証（RADIUS[2]など）が行われているものと想定し、除外している。

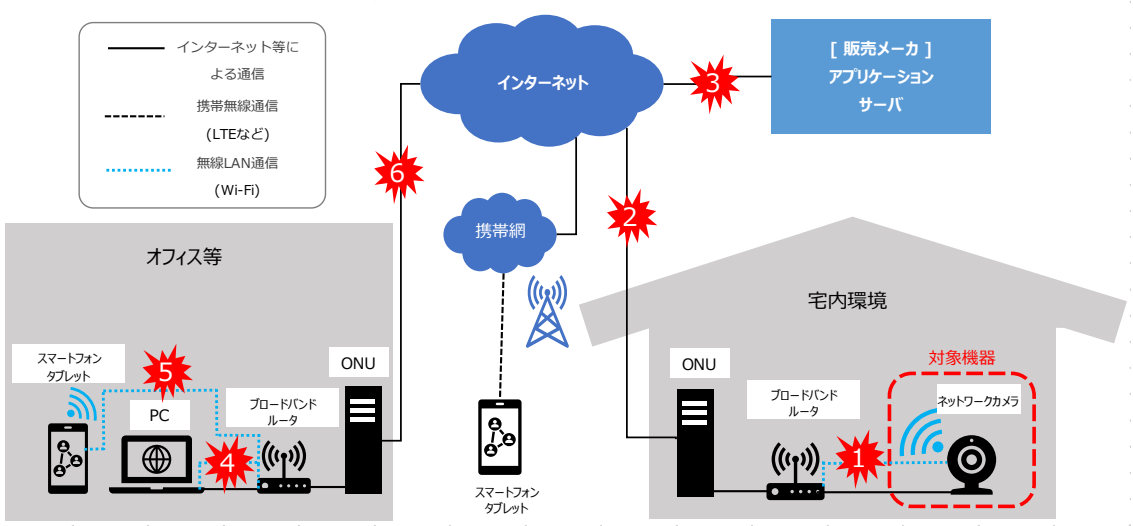

図 3-3 ネットワークカメラシステム構成上の通信経路における攻撃ポイント例

3.3 想定される脅威の分析

3.3.1 脅威分類モデルを活用した脅威の検討

攻撃ポイントの分析を踏まえて、想定される脅威の分析を行う。想定脅威の分析については、マイクロソフトが考案した脅威分類モデル「STRIDE モデル」[3]などが活用でき、前節で抽出した各攻撃ポイントに対して、脅威分類モデルを当てはめていく。脅威分析は要件定義や設計のフェーズで実施され、分析結果を踏まえて具体的なセキュリティ機能の実装方針や事前対策の内容を決定する。

具体的な脅威分類の検討においては、以下のポイントを考慮する必要がある。

- **過去の事例から、各攻撃ポイントにおいて、それぞれの脅威が該当する可能性はあるか**
- **一つの脅威に起因し、複数の脅威が該当する場合には、それぞれを漏れなく抽出できているか**

一つの脅威に起因し、複数の脅威の可能性がある場合であっても、それぞれの脅威に応じて異なるセキュリティ機能の実装や対策が必要となる可能性があるため、該当するものを漏れなく抽出することが必要となる。

一般社団法人重要生活機器連携セキュリティ協議会（以降 CCDS とする）は、「STRIDE モデル」

---

[2] RADIUS：RFC 2865 等で規定されたユーザ認証の認証プロトコル https://tools.ietf.org/html/rfc2865

[3] STRIDE：マイクロソフト提唱による脅威分類モデル https://docs.microsoft.com/ja-jp/security-updates/planningandimplementationguide/19871865

をベースに、IoT 機器等向けに発展させた「CCDS-STRIDE モデル [4]」を提唱している。「CCDS-STRIDE モデル」では、従来の STRIDE モデルに対して 5 つの脅威分類が追加されている。これは IoT 機器等において過去に発生した脆弱性事例を基に脅威分類が追加されたほか、多層防御の考え方に基づき、セキュリティ対策を検討しやすいよう一部の脅威を細分化したものとなる。本別冊では、この「CCDS-STRIDE モデル」を参考に、IoT 機器等への脅威分析にあたってより活用しやすい形での脅威分析モデル（表 3-3）を用いる。なお、「否認」については、オンライントレードなどの分野では脅威として想定されるが、IoT 機器等では脅威事例としては適用できる事例がほぼ見られないことに留意が必要である。本脅威分類モデルをネットワークカメラに適用した例を表 3-4 に示す。

表 3-3 本別冊で用いる脅威分類モデル

脅威名称	英語表記	説明
なりすまし	Spoofing	コンピュータに対し、他のユーザを装うこと
改ざん	Tampering	権限なしでデータを改ざんし、データの完全性を失わせること
否認	Repudiation	ユーザがあるアクションを行ったことを否認し、相手はこのアクションを証明する方法が無いこと
情報漏えい	Informal Disclosure	アクセス権限を持たない個人に情報が公開されること
サービス拒否	Denial of Service	正規のユーザがサーバやサービスにアクセスできないこと
権限昇格	Elevation of Privilege	権限の無いユーザがアクセス権限を得ること
不正アクセス	Unauthorized access	アクセス権限を持たない者にアクセスされること
マルウェア感染	Malware infection	他の機器への汚染源になる。ランサムウェアなどにより業務妨害を受けること
踏み台攻撃	Steppingstone attack	他の機器へ不正アクセス等を行う際の中継地点として使用されること
不正改造(ハードウェア・ソフトウェア)	Tampering with device	不正（違法）なハード、ソフトウェアの改造により、内部データを抜き取りや、脆弱性の要因を組み込まれること

3.3.2 脅威例の検討

各攻撃ポイントに割り当てた脅威分類に対して脅威例を検討する。具体的には、以下のポイントを踏まえて脅威例を検討する。

[4] 一般社団法人重要生活機器連携協議会（CCDS）「IoT 機器に対するリスク分析のガイド」
https://www.ccds.or.jp/certification/document/ccds_risk-analysis-process.pdf

- 対象機器と類似する過去の脆弱性報告事例を参照し、具体的にどのような攻撃が想定されるのか。
- 攻撃方法に対して、影響を受ける対象は機器本体に留まるのか、それとも機器に接続された他の機器やシステムにも影響が発生しうるのか。
- 具体的に攻撃の影響を受ける資産は、どのような資産が想定されるのか。

表 3-4 では、脅威分類モデルを用いて、ネットワークカメラを対象に分析を行った例を示す。脅威例の記載については、CVE レベルの詳細な内容を記載してしまうと、全体としての網羅性を確保することが難しくなるため、以下の記載粒度を参考とされたい。

表 3-4 ネットワークカメラにおける脅威分類、脅威の検討例

対象	攻撃ポイント	脅威分類	影響を受ける守るべき資産	想定される脅威
ネットワークカメラ本体（入出力インタフェース）	A：無線 LAN（Wi-Fi）	不正アクセス	機能・サービス	ネットワークカメラ本体の管理機能やファイル領域にアクセスされ、他の脅威の要因となりえる
		情報漏えい	カメラ映像	映像データの漏えいによるプライバシーの侵害
			録画データ、設定情報、セキュリティ情報	ネットワークカメラ本体の管理機能やファイル領域から、秘匿すべき情報が窃取される
		なりすまし	機能・サービス	認証情報を解析され、攻撃者が利用者になりすまし、機器が利用される
		サービス拒否	機能・サービス	DoS、DDoS 攻撃により、ネットワークカメラの利用が阻害される
		マルウェア感染	機能・サービス	インターネット経由でマルウェア感染につながる。また宅内 LAN 環境に接続された他の機器、システムへ感染源となりえる
		踏み台攻撃	–	インターネット経由でのボットネットの感染などによって、他の機器やシステムに対する攻撃の踏み台につながる
	B：LAN（Ethernet）	不正アクセス	機能・サービス	ネットワークカメラ本体の管理機能やファイル領域にアクセスされ、他の脅威の要因となりえる
		情報漏えい	カメラ映像	映像データの漏えいによるプライバシーの侵害
			録画データ、設定情報、セキュリティ情報	ネットワークカメラ本体の管理機能やファイル領域から、秘匿すべき情報が窃取される
		なりすまし	機能・サービス	認証情報を解析され、攻撃者が利用者になりすまし、機器が利用される
		サービス拒否	機能・サービス	DoS、DDoS 攻撃により、ネットワークカメラの利用が阻害される
		マルウェア感染	機能・サービス	インターネット経由でマルウェア感染につながる。また宅内 LAN 環境に接続された他の機器、システムへ感染源となりえる
		踏み台攻撃	–	インターネット経由でのボットネットの感染などによって、他の機器やシステム

対象	攻撃ポイント	脅威分類	影響を受ける守るべき資産	想定される脅威
				に対する攻撃の踏み台につながる
	C：SD カード、USB インタフェース	情報漏えい	録画データ	SD/USB の盗難、紛失によって、保存された情報漏えいにつながる
		マルウェア感染	機能・サービス	マルウェアが組み込まれた SD/USB との接続により、ネットワークカメラのマルウェア感染につながる
	D：基盤・回路上のポート	不正改造（HW/SW）	プログラムコード	基盤上のデバッグポート（JTAG 端子、UART 端子）を使用し、ソフトウェアの抽出や、解析、改造につながる
システム構成上の通信経路	１：ネットワークカメラとルータ間の無線通信経路	不正アクセス	機能・サービス	無線信号の傍受などの中間者攻撃によって、ネットワークカメラの管理機能やファイル領域にアクセスされ、他の脅威の要因となりえる
		情報漏えい	経路上の通信データ、セキュリティ情報	無線信号の傍受などの中間者攻撃によって、認証情報や通信経路情報を窃取される
		なりすまし	機能・サービス	無線信号の傍受などの中間者攻撃によって、認証情報を窃取され、攻撃者が利用者になりすまし、機器が利用される
	２：宅内 ONU とインターネット間の通信経路	情報漏えい	経路上の通信データ、セキュリティ情報	中間者攻撃による通信経路情報が窃取される
	３：アプリケーションサーバとインターネット間の通信経路	情報漏えい	経路上の通信データ、セキュリティ情報	中間者攻撃による通信経路情報が窃取される
	４：PC（アプリケーションソフト）とルータ間の通信経路	情報漏えい	経路上の通信データ、セキュリティ情報	無線信号の傍受などの中間者攻撃によって、認証情報や通信経路情報を窃取される
		なりすまし	機能・サービス	無線信号の傍受などの中間者攻撃によって、認証情報を窃取され、攻撃者が利用者になりすまし、機器が利用される
	５：スマートフォン（アプリケー	情報漏えい	経路上の通信データ、セキ	無線信号の傍受などの中間者攻撃によって、認証情報や通信経路情報

対象	攻撃ポイント	脅威分類	影響を受ける守るべき資産	想定される脅威
	ション）とルータ間の通信経路		ュリティ情報	を窃取される
		なりすまし	機能・サービス	無線信号の傍受などの中間者攻撃によって、認証情報を窃取され、攻撃者が利用者になりすまし、機器が利用される
	６：オフィス環境 ONU とインターネット間の通信経路	情報漏えい	経路上の通信データ セキュリティ情報	中間者攻撃による通信経路情報が窃取される
上記以外の対象機器に対する物理的な接触	カメラ本体	なりすまし	カメラ映像	虚偽の映像による監視の回避、すり抜け（被写体のなりすまし）
	カメラ本体	サービス拒否	機能・サービス	カメラの自然損壊や、破壊行為による機能の停止

3.4　セキュリティ要求事項の検討

本節では、これまでに実施した脅威分析を基に、セキュリティ機能の実装や事前対策など、対象のIoT 機器等に求められるセキュリティ要求事項（以降、省略し「要求事項」という）の検討を行う。

表 3-5 に IoT 機器等を対象とした代表的なセキュリティ要求事項の例を示す。ユーザ認証、機器認証、デジタル署名については、広義においてアクセス制御に含める文献もあるが、IoT 機器等では製品分野やユースシーンに応じて、各攻撃ポイントの脅威分類に応じて検討する必要があるため、本別冊では、開発者による利便性を考慮し、細分化して追加している。

表 3-5 代表的なセキュリティ要求事項の例 [5]

セキュリティ要求事項（対策名）	機能・目的
セキュア設計	要件定義、設計、コーディングの段階で、脆弱性を作り込まないよう考慮する。
アップデート機能	不具合や脆弱性に対応したソフトウェアやファームウェアへの更新機能や仕組みに対応する。
FW 機能	接続先を IP アドレス・ポート番号で制限する。
サーバ認証	クライアントがサーバを認証することにより、サーバへのなりすましを防止する。
フィルタリング	信頼できない Web サイトへのアクセスを禁止する。また、信頼できないアドレスからのメール受信を拒否する。
IDS/IPS	入出力データを監視し、不正アクセスの検知、抑止を行う。
DoS 対策	DoS（DDoS）攻撃を遮断するための対策を実施する。 可用性を考慮し、機器やシステムに十分なリソースを持たせる。
アンチマルウェア	マルウェアを検知・除去して、マルウェア感染を防止する。
仮想パッチ	ソフトウェア更新等が実施できず、脆弱性を完全に除去できない場合、脆弱性を突いた攻撃を前段にてブロックする。
ユーザ認証	利用者を認証することにより、利用者のなりすましによる脅威を防止する。可能であれば、複数の認証要素を組み合わせた多要素認証技術を採用することが望ましい。
メッセージ認証	通信相手から送信されたメッセージを認証することにより、通信相手へのなりすましによる偽メッセージ送信や、メッセージの改ざんを防止する。
通信路暗号化	データの通信路を暗号化し、通信路上のデータが漏えいしたとしても、無価値化する（攻撃者にとって無意味なものとする）。また、通信路上でのデータの改ざんを検知する。

[5] 独立行政法人情報処理推進機構（IPA）「IoT 開発におけるセキュリティ設計の手引き」を参考に作成
https://www.ipa.go.jp/security/iot/iotguide.html

セキュリティ要求事項（対策名）	機能・目的
データ暗号化	データ自体を暗号化し、仮に蓄積時又は通信時のデータが漏えいしたとしても、無価値化する（攻撃者にとって無意味なものとする）。
データ二次利用禁止	データの目的外利用を禁止し、二次利用先からの漏えいを防止する。
ホワイトリスト制御	あらかじめ許可したプログラム以外の動作を禁止し、マルウェア感染を防止する。
ソフトウェア署名	署名されたソフトウェアの動作のみ許可し、マルウェア感染したソフトウェアや不正改造されたソフトウェアの動作を防止する。
出荷時状態リセット	IoT 機器等を出荷時状態にリセットして、データや出荷後の設定をすべて削除する。
セキュア消去	記録していた場所から復元不可能なようにした上で、データを消去する。
耐タンパ H/W	筐体開封を検知して内部情報を自動消去する等、ハードウェア技術を用いて、内部構造や記憶しているデータの解析を困難とする。
耐タンパ S/W	プログラムやデータ構造の難読化等、ソフトウェア技術を用いて、内部構造や記憶しているデータの解析を困難とする。
遠隔ロック	遠隔操作により IoT 機器等の機能をロックし、第三者による不正利用を防止する。
遠隔消去	遠隔操作により IoT 機器等内のデータを消去し、情報漏えいを防止する。
ログ記録・分析	各種ログを記録し、分析することで、不正アクセスを検知し、何が行われたかを突き止める。
説明書周知徹底	使用上の注意事項を説明書に明記し、使用開始前の利用者の一読を周知徹底する。
アクセス制御	対象へのアクセスを許可、拒否する仕組みにより、なりすましや不正アクセスによる脅威を防止する。
機器認証	利用機器固有の値を用いて、その機器が正しい機器であるかどうかを確認し、なりすましや不正アクセスを防止する。
デジタル署名	送信されたデータが本人ものであることを証明し、データの改ざんやなりすましを検知する。
利用規約による責任範囲の明示	機器やサービスの提供において、免責事項を利用者へ提示し、機器メーカとしての責任範囲を明確化する。また保証期間を設定し、無償修理や交換対応が可能な期間を明示する。

上記のセキュリティ要求事項は、前節までの脅威分析から得られた脅威分類ごとに対応する内容を表 3-6 のように整理することができる。ただし、下記は一般的に該当するものを分類した一例であり、実際の

IoT 機器等においては、実装コストや脅威による影響度、守るべき資産の重要性などを踏まえて、機器ごとに選定すべき事項を検討する必要がある。例えば、比較的低価格で販売されているネットワークカメラや無線 Wi-Fi ルータなどの機器を想定した場合、機器内のデータ暗号化や、耐タンパ HW/SW の実装は現実的とはいえない。一方でビジネスユースとして利用される機器や、医療機器、また金融資産に紐づくデータや個人情報が守るべき資産として含まれる機器については、インシデントが発生した場合の社会的な影響も考慮し、複数の要求事項の対応を検討する必要がある。要求事項を検討にあたっては以下の観点を考慮することが望まれる。

1. **対象機器の販売価格に対して、対応に要するコストは適切か**
2. **守るべき資産に見合った要求事項を選定しているか**
3. **ユースシーンや脅威に見合った要求事項を選定しているか**

表 3-6 脅威モデルとセキュリティ要求事項の対応の例

脅威分類（表 3-3 より）	セキュリティ要求事項 （表 3-5 より）
特定の脅威分類に該当しないもの	セキュア開発
	アップデート機能
なりすまし（偽装）	アクセス制御
	サーバ認証
	ユーザ認証
	機器認証
	メッセージ認証（デジタル署名）
改ざん	通信経路暗号化
	メッセージ認証
否認	デジタル署名
	メッセージ認証
情報漏えい①：通信経路	通信経路暗号化
	アクセス制御
	ユーザ認証
	機器認証
情報漏えい②：ハードウェア・ソフトウェア	データ暗号化
	データの二次利用禁止
	出荷状態リセット
	セキュア消去
	遠隔消去
	耐タンパ―HW
	耐タンパ―SW

脅威分類（表 3-3 より）	セキュリティ要求事項　（表 3-5 より）
サービス拒否	FW（ファイアウォール）機能
	DoS 対策
	電磁波（ジャミング）対策
権限昇格	IDS/IPS
	仮想パッチ
	ログ記録・分析
不正アクセス	FW（ファイアウォール）機能
	アクセス制御
	ユーザ認証
	機器認証
	遠隔ロック
	ログ記録・分析
マルウェア感染	フィルタリング
	アンチウィルス
	仮想パッチ
	ホワイトリスト制御
	ソフトウェア署名
踏み台攻撃	IDS/IPS
	アクセス制御
	ユーザ認証
	機器認証
	ログ記録・分析
不正改造（HW/SW）	ソフトウェア署名
	出荷状態リセット
	セキュア消去
	ソフトウェア署名
	耐タンパ―HW
	耐タンパ―SW
※特定の脅威に紐づかない要求事項	説明書周知徹底
	利用規約による責任範囲の明示

セキュリティ要求事項の検討例として、ネットワークカメラに適用した事例を表 3-7 に示す。一部、家庭用のネットワークカメラとしてはコスト的に対策困難な事項、運用対策が必要な脅威も存在することに留意が必要である。

表 3-7 ネットワークカメラにおける脅威分類、脅威、セキュリティ要求事項の検討例

対象	攻撃ポイント	脅威分類	影響を受ける守るべき資産	想定される脅威例	セキュリティ要求事項
ネットワークカメラ本体（入出力インタフェース）	A：無線 LAN（Wi-Fi）	不正アクセス	機能・サービス	ネットワークカメラ本体の管理機能やファイル領域にアクセスされ、他の脅威の要因となりえる	アクセス制御 ユーザ認証
		情報漏えい	カメラ映像	映像データの漏えいによるプライバシーの侵害	アクセス制御 ユーザ認証
			録画データ 設定情報 セキュリティ情報	ネットワークカメラ本体の管理機能やファイル領域から、秘匿すべき情報が窃取される	出荷状態リセット
		なりすまし	機能・サービス	認証情報を解析され、攻撃者が利用者になりすまし、機器が利用される	アクセス制御
		サービス拒否	機能・サービス	DoS、DDoS 攻撃により、ネットワークカメラの利用が阻害される	※コスト的に対策困難な場合もある
		マルウェア感染	機能・サービス	インターネット経由でマルウェア感染につながる。また宅内 LAN 環境に接続された他の機器、システムへ感染源となりえる	フィルタリング
		踏み台攻撃	–	インターネット経由でのボットネットの感染などによって、他の機器やシステムに対する攻撃の踏み台につながる	アクセス制御
	B：LAN	不正アクセス	機能・サービス	ネットワークカメラ本体の管理機能やファイル領域にアクセスされ、他の脅威の要因となりえる	アクセス制御 ユーザ認証
		情報漏えい	カメラ映像	映像データの漏えいによるプライバシーの侵害	アクセス制御 ユーザ認証

対象	攻撃ポイント	脅威分類	影響を受ける守るべき資産	想定される脅威例	セキュリティ要求事項
			録画データ 設定情報 セキュリティ情報	ネットワークカメラ本体の管理機能やファイル領域から、秘匿すべき情報が窃取される	出荷状態リセット
		なりすまし	機能・サービス	認証情報を解析され、攻撃者が利用者になりすまし、機器が利用される	アクセス制御
		サービス拒否	機能・サービス	DoS、DDoS 攻撃により、ネットワークカメラの利用が阻害される	※コスト的に対策困難な場合もある
		マルウェア感染	機能・サービス	インターネット経由でマルウェア感染につながる。また宅内 LAN 環境に接続された他の機器、システムへ感染源となりえる	フィルタリング
		踏み台攻撃	–	インターネット経由でのボットネットの感染などによって、他の機器やシステムに対する攻撃の踏み台につながる	アクセス制御
	C：SD/USB	情報漏えい	録画データ	SD/USB の盗難、紛失によって、保存された情報漏えいにつながる	説明書周知徹底
		マルウェア感染	機能・サービス	マルウェアが組み込まれた SD/USB との接続により、ネットワークカメラのマルウェア感染につながる	説明書周知徹底
	D：基盤・回路上のポート	不正改造（HW/SW）	プログラムコード	基盤上のデバッグポート（JTAG 端子、UART 端子）を使用し、ソフトウェアの抽出や、解析、改造につながる	※コスト的に対策困難な場合もある
システム構成上の通信経路	１：ネットワークカメラとルータ間の無線通信経路	不正アクセス	機能・サービス	無線信号の傍受などの中間者攻撃によって、ネットワークカメラの管理機能やファイル領域にアクセスさ	アクセス制御 ユーザ認証

対象	攻撃ポイント	脅威分類	影響を受ける守るべき資産	想定される脅威例	セキュリティ要求事項
				れ、他の脅威の要因となりえる	
		情報漏えい	経路上の通信データ セキュリティ情報	無線信号の傍受などの中間者攻撃によって、認証情報や通信経路情報を窃取される	通信経路暗号化
		なりすまし	機能・サービス	無線信号の傍受などの中間者攻撃によって、認証情報を窃取され、攻撃者が利用者になりすまし、機器が利用される	アクセス制御
	２：宅内 ONU とインターネット間の通信経路	情報漏えい	経路上の通信データ セキュリティ情報	中間者攻撃による通信経路情報が窃取される	通信経路暗号化
	３：アプリケーションサーバとインターネット間の通信経路	情報漏えい	経路上の通信データ セキュリティ情報	中間者攻撃による通信経路情報が窃取される	通信経路暗号化
	４：PC（アプリケーションソフト）とルータ間の通信経路	情報漏えい	経路上の通信データ セキュリティ情報	無線信号の傍受などの中間者攻撃によって、認証情報や通信経路情報を窃取される	通信経路暗号化
		なりすまし	機能・サービス	無線信号の傍受などの中間者攻撃によって、認証情報を窃取され、攻撃者が利用者になりすまし、機器が利用される	アクセス制御
	５：スマートフォン（アプリケーション）とルータ間の通信経路	情報漏えい	経路上の通信データ セキュリティ情報	無線信号の傍受などの中間者攻撃によって、認証情報や通信経路情報を窃取される	通信経路暗号化
		なりすまし	機能・サービス	無線信号の傍受などの中間者攻撃によって、認証情報を窃取され、攻撃者が利用者になりすまし、機器が利用される	アクセス制御 ユーザ認証※二段階認証、多要素認証等の高度な認証
	６：オフィス環境 ONU とインター	情報漏えい	経路上の通信データ	中間者攻撃による通信経路情報が窃取される	通信経路暗号化

対象	攻撃ポイント	脅威分類	影響を受ける守るべき資産	想定される脅威例	セキュリティ要求事項
	ネット間の通信経路		セキュリティ情報		
上記以外の対象機器に対する物理的な接触	カメラ本体	なりすまし	カメラ映像	虚偽の映像による監視の回避、すり抜け（被写体のなりすまし）	※機器メーカの対応範囲外なケースが多い
	カメラ本体	サービス拒否	機能・サービス	カメラの自然損壊や、破壊行為による機能の停止	利用規約による責任範囲の明示

4 検証依頼にあたって機器メーカが知るべき検証手法

4.1 機器メーカが知るべき代表的な検証手法

機器メーカが検証サービス事業者に検証を依頼するにあたり、機器に対する代表的な検証手法について、検証の目的・趣旨、検証の流れ、検証結果を踏まえた機器メーカの確認ポイントを解説する。

本別冊で扱う代表的な検証手法と本編の「表 2-1 一般的な機器検証手法」との対応を表 4-1 に示す。各検証手法に関して、検証に必要な情報の収集から実際の検証内容まで体系的に整理している。このため、本編の一般的な機器検証手法に無い手法も一部含まれている。設計文書レビューやソースコード解析等は開発初期に行われる内容であり、本別冊で検証フェーズの対象としている製品開発後の検証であるため本章からは除いている。

表 4-1 本編と別冊の検証手法の対応

別冊		本編	
項	検証手法	分類	検証手法
4.2.1	ハードウェアの情報収集	-	-
4.2.2	デバッグインタフェースの特定	-	-
4.2.3	デバッグインタフェースからのファームウェア抽出	-	-
4.2.4	ファームウェア解析	静的手法	ファームウェア解析
4.3.1	バイナリコードのディスアセンブルとバッファオーバーフロー箇所の特定	静的手法	バイナリ解析
4.3.2	シンボリック実行を利用したバイナリ解析	静的手法	バイナリ解析
4.3.3	APK ファイルのデコンパイル	静的手法	バイナリ解析
4.4.1	ネットワークポートスキャン	動的手法	ネットワークスキャン
4.5.1	ソフトウェア BOM の構築と脆弱性スキャン	静的手法	ファームウェア解析
4.5.2	脆弱性スキャン	動的手法	脆弱性スキャン
4.5.3	脆弱性エクスプロイト	-	-
4.5.4	ハードコーディングパスワードの調査	静的手法	バイナリ解析
4.5.5	パスワードファイルの検証	動的手法	パスワードクラッキング
4.5.6	ネットワークサービスのパスワード検証	動的手法	パスワードクラッキング
4.6.1	ファイルフォーマット・ネットワークプロトコルベースのファジング手法	動的手法	ファジング
4.6.2	コード網羅率を指標とするファジング手法	動的手法	ファジング

別冊		本編	
項	検証手法	分類	検証手法
4.7.1	Ethernet ネットワークパケットキャプチャ	静的手法	ネットワークキャプチャ
4.7.2	Bluetooth パケットキャプチャ	動的手法	ネットワークキャプチャ
4.7.3	アプリケーション通信のパケットキャプチャ	動的手法	ネットワークキャプチャ

機器メーカが検証サービス事業者に検証を依頼する参考例として、家庭用ネットワークカメラを検証対象とした場合に依頼すべき検証手法を表 4-2 に示す。検証ツールが比較的調達しやすく、脆弱性の検出につながる可能性が高い検証手法については、検証を依頼することが望まれる。他方で、一般的な家庭用ネットワークカメラの販売価格を考慮すると、検証にかかるコストと得られる効果が見合わない手法においては、検証の優先度は下がる。

本章の後半では、検証をスムーズに実施する上で機器メーカが事前に用意・提供することが望まれる情報について述べる。

表 4-2 家庭用ネットワークカメラを対象として依頼する検証手法の例

項	検証手法	依頼の優先度
4.2.1	ハードウェアの情報収集	低
4.2.2	デバッグインタフェースの特定	低
4.2.3	デバッグインタフェースからのファームウェア抽出	低
4.2.4	ファームウェア解析	低
4.3.1	バイナリコードのディスアセンブルとバッファオーバーフロー箇所の特定	低
4.3.2	シンボリック実行を利用したバイナリ解析	低
4.3.3	APK ファイルのデコンパイル	低
4.4.1	ネットワークポートスキャン	高
4.5.1	ソフトウェア BOM の構築と脆弱性スキャン	高
4.5.2	脆弱性スキャン	高
4.5.3	脆弱性エクスプロイト	低
4.5.4	ハードコーディングパスワードの調査	高
4.5.5	パスワードファイルの検証	高
4.5.6	ネットワークサービスのパスワード検証	高
4.6.1	ファイルフォーマット・ネットワークプロトコルベースのファジング手法	低
4.6.2	コード網羅率を指標とするファジング手法	低
4.7.1	Ethernet ネットワークパケットキャプチャ	高

項	検証手法	依頼の優先度
4.7.2	Bluetooth パケットキャプチャ	低 [6]
4.7.3	アプリケーション通信のパケットキャプチャ	高

4.2 ハードウェアハッキング・ファームウェア解析

機器の脆弱性やセキュリティ上の不備を見つけるためには、ブラックボックスの状態で検証を行うよりも、機器の内部の情報を参照した上で検証を行った方がより有効な情報が見つかることが多い。内部の情報を取得する上で、ハードウェア上のチップやデバッグインタフェースから機器のストレージやコンソールにアクセスし、ファームウェアやファイルシステムにアクセスする方法がある。本節では、このようなハードウェアを攻撃ポイントとした検証手法について解説する。

4.2.1 ハードウェア情報の収集

（1） 検証の目的・趣旨

本手法の目的は、機器が利用しているハードウェアやその仕様の公開情報等から検証に関連する情報を収集することである。本手法による検証の結果として次のような情報が得られる。

- 機器の基板上から特定・推定可能な情報
  - チップメーカ、型番
  - デバッグインタフェース
- 上記に関連した、ハードウェア関連の公開情報やデータシート

本手法で収集した情報は、後述のデバッグインタフェースの特定、バイナリ解析等の検証工程の短縮、脆弱性の攻撃難易度のリスク評価等に利用可能である。これらの情報は、基板から設計・開発するメーカにとって既知の情報も含まれるが、チップや OEM 等の部品を利用する場合に、ハードウェア上から第三者がどのような情報を収集可能か、評価する材料となる。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

ハードウェアに関する情報を収集する際には、主に次の情報源がある。

- 機器の基板上のチップ形状、シルク
- インターネット上の公開情報（利用しているチップのメーカ、型番、データシート）
- 機器の仕様書

機器の基板上のチップ形状からは、チップの大きさやピンの本数からチップの種類やメーカが推定可能である。また、チップに印字されているプリント文字から型番やメーカ情報等の詳細な情報が特定できる場合もある。

インターネット上の情報としては、検証対象の機器が搭載しているチップのメーカや型番、データシート等が該当する。当該情報について機器メーカが公開している事例はあまり無いが、第三者がハッキングし

[6] Bluetooth の機能を実装している家庭用ネットワークカメラは少ないため、優先度を「低」とした。

た結果として公開されている事例がある。販売前の機器であっても、旧式の機器が分析されているケースもある。このような情報は、海外のハッキング用 Web サイトでまとめられている場合や、国内でも個人が解析した結果をブログ記事として公開している事例がある。また、チップのメーカや型番が特定できれば、チップメーカが公開しているデータシートを参照することも可能となる。

以上のように機器の基板を観察することで、チップの形状やシルクからチップメーカや型番を特定し、関連するデータシートの参照にもつながりうる。本検証結果は脆弱性に直接該当する情報ではないが、第三者も収集可能な情報であり、後述のファームウェア抽出や脆弱性の特定にもつながる可能性があることを機器メーカは理解すべきである。

検証結果を踏まえて機器メーカが確認するポイント

- 検証対象機器のハードウェアから収集可能な情報を把握する
- 各チップのデータシート等の公開状況を把握する
- 上記の情報が検証対象機器への攻撃に影響する内容であることを理解する

4.2.2 デバッグインタフェース（JTAG, UART）の特定とアクセス可否の確認

（1） 検証の目的・趣旨

機器のデバッグインタフェースが有効な場合、このインタフェース経由でファームウェアの抽出や、シリアルコンソールからログやシェルへアクセスできる可能性がある。例えば、UART インタフェース経由でシリアルコンソールにアクセスすると、機器の動作ログ情報を確認できたり、シェルにログインして機器を内部から操作できたりすることがある。本検証手法では、このようなデバッグインタフェースを特定し、実際に利用可能かどうかを確認する。

本手法による検証の結果として次のような情報が得られる。

- デバッグインタフェースへの接続情報
  - ピン配置
  - ボーレート、パリティ、標準ビット数、ストップビット値、フロー制御
- デバッグインタフェース経由で取得できる情報
  - シリアルコンソールに出力されるログ情報

検証サービス事業者は本手法によりデバッグインタフェースへの接続情報を特定するが、第 4.2.1 項の手法でデータシートを入手できた場合は、デバッグインタフェースのピン配置やボーレート情報等を参照し、工程を省略できる場合がある。配置の情報が無い場合には基板の配置を観察し、マルチメータやオシロスコープ・ロジックアナライザでテストすることで配置を特定することができる。

また、シリアルコンソールに接続できた場合には、検証対象機器のログ情報が出力されることがあり、デバイス情報や稼働するソフトウェア・サービス等、第 4.3 節以降の手法で活用できる情報が含まれることがある。

（2）検証の流れと検証結果を踏まえた機器メーカの確認ポイント

本手法は、デバッグインタフェースの箇所を特定し、デバッグ用シリアル通信機器を接続してデバッグインタフェースの利用可否を確認する流れとなっている。本手法の流れは次のとおりである。

1. 基板上のデバッグインタフェース箇所の推定
2. 基板上のグランド（GND）箇所の特定
3. 基板上のデバッグインタフェース配置の特定
4. デバッグインタフェースの挙動確認

手順 1 では、基板上にあるデバッグインタフェースの接続箇所を推定する。基板上に TX や RX 等の各インタフェースに応じた印字を確認する方法やピン配置の特徴から推定する方法もある。UART の場合は、図 4-1 のように 3～4 つの穴（スルホール）が並んでいるパターンが存在する。

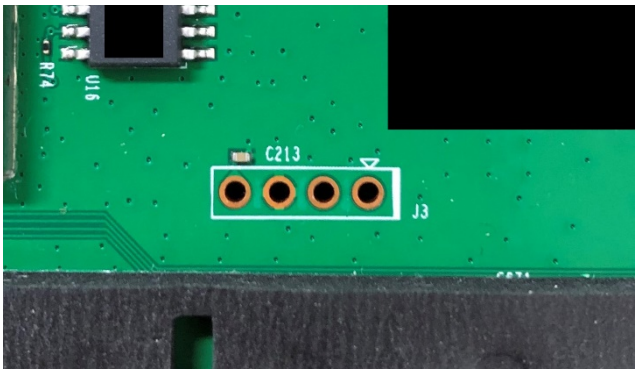

図 4-1 UART インタフェース接続用のスルホール

手順 2 では、マルチメータ、サーキットテスタの導通チェックによりグランドの箇所を特定する。デバッグインタフェースの配置にはグランドが含まれており、手順 3 のデバッグインタフェース箇所の特定に必要である。

手順 3 では、デバッグインタフェースの配置を特定する。例えば、UART は 3 線方式で、各ピンにデータ送信や受信の役割があるため、基板のどのピンがどの役割かを特定する。主な特定方法として、各ピン・スルホールにオシロスコープやロジックアナライザ等を接続し、電圧の変化から判定する方法がある。また、ピンの特定の自動化・補助ツールを利用することで効率的にデバッグインタフェースの配置を特定することができる。このツールの例として JTAGulator がある。

手順 4 では、図 4-2 のように、特定したデバッグインタフェースに対応したアダプタツールを接続し、デバッグインタフェースが実際に有効かどうかを確認する。デバッグインタフェースに対応したアダプタツールには、Attify Badge や Buspirate 等のツールがある。これらのツールは UART の他にも JTAG デバッグインタフェースや、SPI や I2C のようなシリアル通信インタフェースに対応している。

デバッグインタフェースからは、シリアルコンソールを経由して動作ログやシェルへアクセスされる可能性がある。機器メーカは、特に OEM 製品やサードパーティの開発キットを利用する機器の場合、本検証によりデバッグインタフェースの有効性を確認し、デバッグ機能を利用してメモリ情報参照やファームウェア抽出されるリスクについて理解すべきである。

検証結果を踏まえて機器メーカが確認するポイント

- 検証対象機器の仕様上、デバッグインタフェースを許可しているかどうかを確認する
- デバッグインタフェースが有効どうかを確認する
- デバッグインタフェースから収集可能な情報や利用可能な機能を確認する

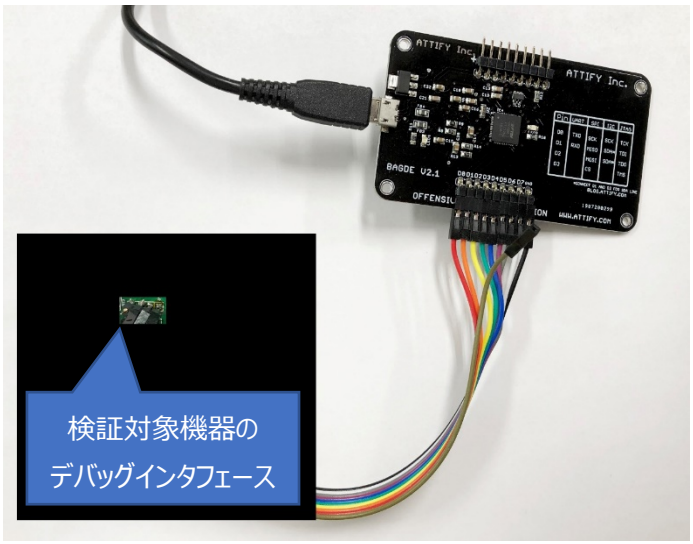

図 4-2 デバッグインタフェースのアダプタと検証対象機器の接続例

4.2.3 デバッグインタフェースからのファームウェア抽出

（1） 検証の目的・趣旨

ファームウェアには検証対象機器のプログラムファイルや設定情報等、脆弱性の診断に有用な情報が含まれている。開発メーカ以外の第三者がファームウェアを入手する方法として、サポートサイトで公開されているアップデートファイルをダウンロードする方法、デバッグインタフェースから抽出する方法、EEPROM やフラッシュメモリ等のストレージから直接抽出する方法等がある。本節では、デバッグインタフェースからファームウェアを抽出する手法について解説する。

（2）検証の流れと検証結果を踏まえた機器メーカの確認ポイント

デバッグインタフェースを経由してファームウェアを抽出する代表的な手法として、次の手法がある。

- UART インタフェース経由
  - U-Boot[7]のコマンドシェルで RAM のデータ出力関連コマンドを実行
  - ログインシェルでデータ出力関連コマンドを実行
- JTAG インタフェース経由

[7] 組み込みシステムの機器で広く利用されている OSS のブートローダ。

➢ フラッシュメモリ・RAM のデータ出力関連コマンドを実行

UART インタフェース経由の手法では、シリアルコンソールに接続後、U-Boot やシェルのコンソールが利用可能な場合に、フラッシュメモリのデータを出力する関連コマンドを実行してファームウェアイメージを抽出する。JTAG インタフェース経由の手法では、JTAG のデバッグ用コマンドを実行し、フラッシュメモリ上のデータのダンプや、メモリ上にロードしたファームウェアイメージを抽出する。これらの検証の流れと検証結果の確認ポイントを以下で紹介する。

- **UART インタフェース経由の手法：U-Boot のコマンドシェルで RAM のデータ出力関連コマンドを実行**

検証対象機器がブートローダとして U-Boot を利用しており、U-Boot のコマンドシェルが有効な場合、UART 経由で U-Boot 関連のコマンドを実行できる。

- 1. ホスト PC を UART インタフェースに接続
- 2. 検証対象機器を起動
- 3. キーボードで自動起動を中断し、U-Boot のコマンドシェルにアクセス
- 4. フラッシュメモリから RAM へファームウェアイメージをロード
- 5. RAM の内容を出力し、ファームウェアイメージのデータを取得

U-Boot の標準コマンドには RAM やフラッシュメモリのデータを出力するコマンドがあり、この内容をコンソールに出力してホスト PC 上でバイナリに変換する。

- **UART インタフェース経由の手法：ログインシェルでデータ出力関連コマンドを実行**

UART 経由でログインシェルにアクセスできる場合、次の流れでファームウェアを抽出可能である。

- 1. ホスト PC を UART インタフェースに接続
- 2. 検証対象機器を起動
- 3. ログインシェルにアクセス
- 4. ストレージデバイスをダンプ
- 5. ダンプしたデータをホスト PC へ転送

UART インタフェースへの接続までは上記の手法と同様だが、ログインシェルに接続後、ファームウェアイメージを取得することが違いである。bash 等のログインシェル経由でコマンドを実行できるため U-Boot のシェル機能よりも自由度が高く、ファームウェアを取得する方法は多岐に亘る。例えば、dd コマンドでストレージデバイスの内容を出力し、検証対象機器の外部ストレージに出力する方法や、コマンドの出力結果をネットワーク経由でデータを転送する方法がある。また、hexdump コマンド等でファームウェアイメージを ASCII フォーマットに変換してシリアルコンソールに出力し、出力結果をバイナリに再変換して元に戻す方法がある。

- **JTAG インタフェース経由：フラッシュメモリ・RAM データ出力関連コマンドを実行**

JTAG にはフラッシュメモリや RAM のデータを操作するコマンドがあり、RAM 上にロードしたファームウェアイメージを JTAG インタフェース経由で取得可能な場合がある。検証対象機器のアーキテクチャや JTAG

アダプタに合わせた通信用ソフトウェアが必要であるが、ARM や Cortex 等のアーキテクチャや各種 JTAG アダプタに対応した OpenOCD や UrJTAG が OSS として利用可能である。OpenOCD では、メモリ操作のコマンド（md，dump_image 等）で RAM にロードされたファームウェアイメージを出力する方法や、フラッシュメモリ操作のコマンド（flash）でフラッシュメモリ上のデータを出力する方法等がある。

ただし、CPU のアーキテクチャ、RAM やフラッシュメモリのメモリ配置等、JTAG 通信を実現するための設定項目がある。OpenOCD ではサポートしているハードウェアや JTAG アダプタの設定ファイルが公開[8]されており、これらの設定ファイルを利用可能である。設定ファイルの公開情報が無い場合には、チップのデータシートを参考に検証者が作成する必要がある。

これら手法により、UART や JTAG デバッグインタフェース経由でファームウェアを抽出可能である。ファームウェアを実際に抽出可能な場合、ファームウェアからの脆弱性の特定や不正なファームウェアの作成につながる可能性があるため注意が必要である。

検証結果を踏まえて機器メーカが確認するポイント

- デバッグインタフェースを経由してファームウェアが抽出可能かどうかを確認する
- 抽出されたファームウェアが第三者による脆弱性の特定や不正なファームウェアの作成に利用されるリスクを確認する

4.2.4 ファームウェア解析

（1） 検証の目的・趣旨

ファームウェア解析では、第 4.2.3 項のファームウェア抽出によって得られたファームウェアイメージを解析し、イメージ内に含まれるデータや情報を調査する。本手法による検証では主に次の情報が得られる。

- ファームウェアイメージの圧縮・暗号化の有無情報
- ファームウェアイメージに含まれるファイル・ファイルシステム

一般的に、ファームウェアイメージは複数のファイルデータが一つのバイナリデータにまとめられており、中には暗号化しているケースもある。圧縮や暗号化の有無についてはエントロピー解析[9]により推定できる。また、ファームウェアにはプログラムファイルや設定ファイルが含まれており、脆弱性の検出や認証情報の特定等につながる可能性がある。本節ではファームウェアの圧縮・暗号化の有無の推定とファイルシステムの抽出に焦点を当てた手法について述べ、ファイルシステム内に含まれるプログラムファイルや設定ファイルの内容の解析については第 4.3 節以降で紹介する。

（2）検証の流れと検証結果を踏まえた機器メーカの確認ポイント

ファームウェアのイメージファイルは通常一つのバイナリデータとなっており、ファイルマネージャやテキストエディタでは内部を参照することができない。バイナリエディタを利用した検証者による手動の分析は可能で

[8] OpenOCD の標準設定ファイルリポジトリ https://sourceforge.net/p/openocd/code/ci/master/tree/tcl/

[9] データの情報量（エントロピー）を解析して定量化する手法であり、圧縮や暗号化されたデータの場合は高い値を示す。

はあるが、ファームウェアのような大きいサイズの解析は困難である。ファームウェア解析を補助する代表的なツールとして binwalk がある。binwalk は主に次の機能がある。

- エントロピー解析
- ファイルの抽出

エントロピー解析では、ファームウェアイメージの圧縮又は暗号化の有無の推定に利用する。エントロピーは情報理論の分野ではデータの規則性やランダムさの指標であり、binwalk では入力データ（ファイル）のエントロピー値を計算する。入力データが圧縮又は暗号化されている場合にエントロピー値が高い値を示すため、ファームウェアイメージの圧縮・暗号化有無の推定に利用できる。

ファイルの抽出では、ファームウェアイメージに含まれるファイルを抽出する。binwalk は入力されたデータを対象に、バイナリフォーマットの特定のパターン（シグネチャ）に合致するデータを特定し、バイナリデータ内のどの位置にどのようなデータが含まれるのか出力する。この出力結果を基に dd コマンドやバイナリエディタでバイナリデータからファイルを抽出することが可能である。Linux 系の組み込みデバイスを中心としたファームウェアのファイルシステムには一般的に Squash ファイルシステムが利用されるケースが多く、これらのファイルシステムのデータを展開し、最終的にプログラムデータや設定ファイル等のファイルを参照することができる。

検証結果を踏まえて機器メーカが確認するポイント

- ファームウェアを暗号化している場合、エントロピー解析結果として適切な結果となっているかを確認する
- ファームウェアから収集可能な情報を確認する

4.3 バイナリ解析

本検証では、ファームウェアに含まれるプログラムファイルを解析し、脆弱性につながりうる情報や機微な情報が含まれていないか解析する。解析の手法として、バイナリプログラムをアセンブラコードに変換するディスアセンブル（逆アセンブル）や C 言語や Java 言語等のレベルまで変換するデコンパイル（逆コンパイル）を行い、出力された変換後のコードを確認していく。

4.3.1 バイナリ解析とバッファオーバーフロー脆弱性の特定

（1） 検証の目的・趣旨

本検証の目的は、バイナリコードのディスアセンブル結果を基に、脆弱性の検出や攻撃者に有益な情報が無いか解析することである。本検証を補助するツールとして、IDA Pro や Ghidra、radare2 等がある。これらのツールはディスアセンブル機能だけでなく、変換後のコードを基に、プログラム構造の自動解析、デバッグによる挙動の確認、ハードコードされた文字列や参照しているファイルパス情報を取得する機能等が存在し、これらの機能を活用して解析をスムーズに進めることができる。本節では脆弱性検出の例としてバッファオーバーフロー脆弱性の特定の流れを紹介する。

（2）検証の流れと検証結果を踏まえた機器メーカの確認ポイント

ディスアセンブル後の解析でバッファオーバーフロー脆弱性を特定するアプローチの一つを紹介する。

1. バイナリコードをディスアセンブルする
2. memcpy, strcpy 等のメモリ操作関連の関数の利用箇所を特定する
3. メモリ操作用バッファのサイズの確認や、コピーサイズ指定の変数の確認

バイナリコードのディスアセンブルの結果として主に次の情報が得られる。

- ディスアセンブル後のアセンブリコード
- バイナリコードのアーキテクチャ
- 参照している共有ライブラリ・関数
- ハードコードされた文字列データ
  - 設定ファイルパス
  - プログラムで出力されるメッセージ

先述の補助ツールでは、バイナリコードのディスアセンブル後にプログラムの制御フローや参照ライブラリ等を自動で解析できる。これにより、検証者はプログラム内で利用されている関数の参照、関数がどこから呼び出されているかを逆参照することができる。

バッファオーバーフロー脆弱性は、主にメモリ操作を行う制御箇所に潜在しているため、メモリ操作を行う memcpy や sprintf 等の関数の呼び出し箇所からバッファオーバーフロー脆弱性の箇所を特定していくアプローチが存在する。このアプローチでは、メモリ操作関連の関数の引数に渡されるバッファやバッファサイズ指定のための変数への参照を辿っていく。最終的にサイズ指定の変数がプログラムへの入力値に依存する場合、バッファオーバーフローが起こる可能性がある箇所として推定できる。例えば、検証対象機器へのデータ入力があり、当該データフォーマットにおいてペイロード長を指定するような構造になっている場合、検証対象機器のプログラムではこのペイロード長の値のサイズ分のデータをバッファへコピーする処理がある。このとき、ペイロード長の値は機器外部から操作可能なため、攻撃者が意図的に値を操作してバッファオーバーフローにつながる可能性がある。

以上のように、バイナリコードのディスアセンブル及びバイナリ解析により脆弱性が特定されることがある。特にバッファオーバーフロー脆弱性が起こりやすい点についてはセキュアコーディングの基礎としても一般的となっている。機器メーカとしては、本検証で得られた情報からセキュリティアップデートのパッチ開発や、開発プロセスへのフィードバックとしてセキュアコーディング時の注意事項として活用することも可能である。

検証結果を踏まえて機器メーカが確認するポイント

- バイナリコードに脆弱性やバックドアとなる機能や処理が含まれていないか確認する
- バイナリコード内の文字列データにパスワードやアクセストークン等の機密性の高い情報が含まれていないか確認する

4.3.2 シンボリック実行を利用したバイナリ解析

（1） 検証の目的・趣旨

第 4.3.1 項ではディスアセンブル後のコードからバッファオーバーフロー脆弱性の可能性がある箇所を特定する流れを紹介した。この例では入力データがバイナリコード内の特定の箇所に依存するかどうかを解析する流れとなっていた。ただし、このバイナリコード内のプログラムの流れを手動で追うことは時間がかかり検証にかかる工数が増えてしまう場合がある。これに対し、バイナリコードを静的又は動的に解析する手法により解析の自動化が可能である。本節では、バイナリコード内のデータフローの解析や、バイナリコード内の特定のアドレスに到達する入力データを解析する手法として、シンボリック実行を紹介する。

（2）検証の流れと検証結果を踏まえた機器メーカの確認ポイント

シンボリック実行は、バイナリ内の実行パスの抽出やその実行パスに到達するための入力値の特定を目的とした解析手法である。シンボリック実行による解析用ライブラリとして Triton や angr があり、解析のためにコーディングが必要である。これらのライブラリはシンボリック実行の他にも、バイナリコードのデータ制御フローの解析やテイント解析 [10]の機能を備えているものもある。

Triton を利用したシンボリック実行の主な流れを次に示す。

1. バイナリコード情報の設定
2. バイナリコードのロード
3. シンボルの定義
4. シンボリック実行による解析

バイナリコード情報の設定では、バイナリコードのアーキテクチャや攻撃ポイント等を設定する。これらの情報にはライブラリの実行に必要な情報が含まれるが、前節の IDA Pro や Ghidra を利用して情報収集や解析が可能である。次に、解析対象となるバイナリコードをロードする。対象のバイナリコードはファームウェア解析結果のバイナリコードを用いることや、その一部を切り取って解析することも可能である。シンボルの定義では、レジスタやメモリの所定のアドレス等をシンボルとして定義する。シンボリック実行の解析結果として、定義されたシンボルに対する操作が記録される。この操作の記録から、所定の実行パスに到達する入力値を計算することができる。

上記の内容はバイナリコードの所定の実行パスへの入力値を求める例であり、その他にも実行される関数の列挙や参照されるメモリアドレスの一覧作成等、目的に応じて自由にプログラムできる。

検証結果を踏まえて機器メーカが確認するポイント

- シンボリック実行により解析できる入力値、実行パスを確認する

[10] バイナリコードにデータを入力した際に当該データに関連するデータフローを解析する手法。

4.3.3 APK ファイルのデコンパイル

（1） 検証の目的・趣旨

検証対象機器の制御や設定等に対応したモバイルアプリケーションには、検証対象機器が利用する通信プロトコルや制御命令データ、パラメータ値等の情報が含まれており、これらの情報を基にした不正操作や情報漏えいの可能性がある。本検証ではモバイルアプリケーションからこのような情報を収集可能かどうか確認することを目的とし、バイナリ解析を行う。特に、Java 言語で開発されている Android のモバイルアプリケーションでは、バイナリコードからソースコードへデコンパイルする手法が確立されている。本節では、Java 言語で開発された Android 用モバイルアプリケーションデータである APK ファイルのデコンパイルについて紹介する。

（2）検証の流れと検証結果を踏まえた機器メーカの確認ポイント

APK ファイルをデコンパイルする代表的な流れは次のとおりである。

1. APK ファイルを展開する
2. 展開後に含まれる DEX ファイルを JAR ファイルへ変換する
3. JAR ファイルに含まれる CLASS ファイルを展開する
4. CLASS ファイルを JAVA ファイルへ変換する

APK ファイルはデータ圧縮フォーマット ZIP をベースにファイル構造が定義されているファイルフォーマットであり、ZIP の展開用ソフトウェアで展開可能である。APK ファイルには、マニフェストファイル [11]や各種ライブラリ等のデータが格納されている。このうち classes.dex ファイル（DEX ファイル）にプログラムのバイナリコードが格納されており、DEX ファイルを Java 言語のソースコードへ変換する。これらの展開や変換を行うツールとして、jadx や apktool がある。各ステップの展開・変換後のファイルから得られるハードコードされた情報やソースコードを基に、ハードコードされた情報やプロトコルに関する制御情報を解析していく。これら解析手法については第 4.5 節以降で紹介する。

APK ファイルのデコンパイルは C 言語や C++言語等のディスアセンブルやデコンパイルよりもソースコードに近い形で結果が得られるため、デコンパイル後にハードコードされた情報やソースコード中の制御フロー等が第三者によって解析されやすいことを機器メーカは理解しておくべきである。

検証結果を踏まえて機器メーカが確認するポイント

- デコンパイル後のコードに脆弱性やバックドアとなる機能や処理が含まれていないか確認する
- デコンパイル後のファイルにパスワードやアクセストークン等の機密性の高い情報が含まれていないか確認する

4.4 ネットワークスキャン

[11] アプリケーションのタイトルやバージョン、アクセス権限等の情報を定義するファイル。

4.4.1 ネットワークポートスキャン

（1） 検証の目的・趣旨

ネットワークポートスキャンでは、検証対象機器が待ち受け状態の TCP/UDP サービスポートを確認する。本検証の目的は、機器の仕様上不要なサービスや意図しないサービスが動作していないかを確認することである。問題となる例として、デバッグ用に使用していたサービスポートが開いたままになっており、機器への不正アクセスを許してしまうケースがある。

また、ネットワークポートスキャンのもう一つの目的は、待ち受け状態の TCP/UDP サービスポートに接続して収集可能な情報を確認することである。例えば、TCP80 番ポートで HTTP を利用するサービスが稼働している場合、通信のレスポンスデータに Web サーバの名称やバージョン情報が含まれていることがある。このような情報はフィンガープリントと呼ばれており、脆弱性検出へのヒントとなる可能性がある。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

ネットワークポートスキャンの流れは次のとおりである。

1. 検査のホスト PC と検証対象機器を同一ネットワークに接続する
2. ポートスキャンツールを利用してポートスキャンを実施する

ポートスキャン時の留意点として、Wi-Fi ルータのように検証対象機器が別々のネットワークに対するインタフェースを持つ場合、それぞれのネットワークインタフェースに対してポートスキャンを実施する必要がある。また、ポートスキャンの代表的なツールに nmap があるが、標準設定ではよく利用されるサービスポートのみを検査する設定となっている。組み込み機器では IANA[12]で定義されている標準的なサービスポート以外で待ち受け状態のサービスポートを開放している機器も多いため、すべてのサービスポートを対象としたポートスキャンの実施が推奨される。本検証による確認ポイントとして、待ち受け状態のサービスポートは不正アクセスにつながるリスクを高めることから、検証対象機器で待ち受けているサービスポートを把握し、当該サービスポートに対するセキュリティリスクを評価することが望ましい。

検証結果を踏まえて機器メーカが確認するポイント

- 検証対象機器が待ち受けているサービスポートが必要なポートかどうか確認する
- 検証対象機器のファイアウォール機能等で適切にフィルタリングされていることを確認する

4.5 既知脆弱性の診断

既知の脆弱性はエクスプロイト手法が公開されている場合があり、脆弱性を悪用して攻撃を受ける可能性が高い。本節では検証対象機器に内在する既知の脆弱性を診断する手法について解説する。

[12] Internet Assigned Numbers Authority, IP アドレスやドメイン名、ネットワークサービスポート番号等を管理する国際的な組織。

4.5.1 ソフトウェア BOM の構築・脆弱性スキャン

（1） 検証の目的・趣旨

本検証の目的は、検証対象機器が利用しているライブラリやコンポーネントに既知の脆弱性が無いかを確認することである。既知の脆弱性はその情報や攻撃方法が広く公開されており、サイバー攻撃の対象となる可能性が高い。既知の脆弱性の有無は、検証対象機器が利用しているソフトウェアとその脆弱性情報を照合することで確認でき、利用しているソフトウェアの一覧としてソフトウェア BOM を作成する。ソフトウェア BOM は利用しているソフトウェアの部品表であり、ソフトウェア名称、バージョン、ライセンス等の情報を管理する。本検証の結果として、検証対象機器のソフトウェア BOM と関連する脆弱性情報が得られる。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

ソフトウェア BOM の作成手法として、主に次の手法がある。

1. 仕様書や設計書を基に作成する手法
2. 検証対象機器に対してネットワークスキャンを実施し、フィンガープリント情報から作成する手法
3. ファームウェアイメージをツールで自動解析して作成する手法

一般的に検証を依頼する場合には、2.又は 3.の手法により作成することが多い。作成結果の精度として、検証時に得られる情報の違いのため 2.のネットワークスキャンよりも 3.のファームウェア解析による手法の方が正確な情報を得られることが多い。

本検証の結果として、検証対象機器で利用されているソフトウェアやそのバージョン、ライセンス情報が得られる。これらの検証結果と脆弱性データベースを照らし合わせ、既知の脆弱性が含まれるかどうかを確認する。脆弱性データベースの例として、米国国立標準技術研究所（NIST）が公開している NVD[13]や、JPCERT/CC・IPA が公開している JVN[14]等がある。また、セキュリティ検証を目的としたソフトウェア BOM 作成ツールには、ソフトウェア BOM と脆弱性の公開情報と照合する機能があり、この機能を用いて脆弱性の照合に係る処理を自動化できる。このようなソフトウェア BOM 作成ツールの例として、VDOO Vison や Centrifuge、FACT 等がある。

本検証により機器メーカは検証対象機器に潜在する脆弱性を確認できる。脆弱性の深刻度によっては対策やユーザへの注意喚起等が早急に必要な内容もあるため、深刻度の高い脆弱性が検出された場合には逐次報告するよう検証サービス事業者に依頼する。

以上のように、検証対象機器のソフトウェア BOM を作成し、利用しているソフトウェアやバージョン、ライセンス情報を本検証により確認可能である。また、ソフトウェア BOM と脆弱性情報を照合することで、検証対象機器に潜在する脆弱性について確認することができる。

検証結果を踏まえて機器メーカが確認するポイント

[13] NATIONAL VULNERABILITY DATABASE https://nvd.nist.gov/

[14] Japan Vulnerability Notes https://jvn.jp/

- ソフトウェア BOM を構築し、検証対象機器が利用しているソフトウェア、バージョン、ライセンス情報を確認する
- ソフトウェア BOM と脆弱性情報を照合し、検証対象機器に潜在する脆弱性を確認する
- 深刻度の高い脆弱性が検出された場合には、逐次報告するよう検証サービス事業者に依頼する
- 脆弱性に対するソフトウェアの修正のほか、緩和策について検討する

4.5.2 ネットワーク経由の脆弱性スキャン

（1） 検証の目的・趣旨

本検証の目的は、検証対象機器に対して脆弱性スキャンを行い、既知の脆弱性が顕在化しているかどうかを確認することである。本節では、ネットワークを攻撃ポイントとする脆弱性スキャンによる検証手法を紹介する。本検証では、主にネットワーク経由で取得できる情報と脆弱性の公開情報を照合して、脆弱性の有無を推定する。本検証では、検証ツールがアクセスできる内容が脆弱性検出につながるため、検証対象機器のアカウント情報（ID、パスワード）等の情報を検証者に提供することで、より多くの範囲を対象とした検証を実施できる。

本検証の結果として、検証対象機器の脆弱性情報が得られる。また、利用するツールや脆弱性診断サービスによっては、実際の攻撃手法（エクスプロイト）や脆弱性への対策・緩和策についての情報も同様に報告されることが多い。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

本検証の主な流れは次のとおりである。

1. ネットワークスキャンによる待ち受けサービスポートの確認
2. 待ち受けサービスポートに対応するプロトコル通信を行い、対象サービスからの情報収集
3. 情報収集結果と脆弱性情報を照合

まず、ネットワークスキャンにより検証対象機器がネットワーク経由で待ち受けているサービスポートを確認する。次に、待ち受けサービスポートに対応するプロトコル通信を行う。例えば、Web 管理機能の提供のための HTTP サービスポートの場合、HTTP クライアントで接続して HTTP のプロトコルバージョン、Web サーバの名称・バージョン等を収集する。最後に、この収集結果と脆弱性の公開情報を照合し、脆弱性の有無を推定する。脆弱性スキャンツールによっては、脆弱性の攻撃パターンの通信を試行し、未知の脆弱性を検出するものもある。本検証を行うツールの例として、Nessus や Greenborne Vulnerability Manager（OpenVAS）等がある。

本検証により脆弱性が含まれるコンポーネントと脆弱性情報が得られる。第 4.5.1 項の手法と同様に、深刻度の高い脆弱性が検出された場合には逐次報告するよう検証サービス事業者に依頼する。

検証結果を踏まえて機器メーカが確認するポイント

- 脆弱性スキャン結果を確認し、検証対象機器に潜在する脆弱性を確認する

- 深刻度の高い脆弱性が検出された場合には、逐次報告するよう検証サービス事業者に依頼する
- 脆弱性に対するソフトウェアの修正のほか、緩和策について検討する

4.5.3 脆弱性エクスプロイト

（1） 検証の目的・趣旨

第 4.5.1 項や第 4.5.2 項の脆弱性スキャンにより、検証対象機器の脆弱性情報が得られる一方で、誤検知の場合や設定やアーキテクチャの違いにより、実際には脆弱性の影響を受けない場合がある。本検証では、報告された脆弱性を実際に攻撃（エクスプロイト）して挙動を確認することで、脆弱性の影響を確認することを目的とする。本検証の結果として、脆弱性情報の実際の影響の有無を確認することができる。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

第 4.5.1 項や第 4.5.2 項の脆弱性スキャンで得られた脆弱性情報を基に、具体的な攻撃手法を設計し、検証対象機器に対して試行する。脆弱性への攻撃手法は、エクスプロイト情報としての公開や、脆弱性の概念実証（PoC：Proof of Concept）としてテスト用のスクリプトが公開されているケースがある。例えば、エクスプロイト情報が公開されている EXPLOIT DATABASE[15]において脆弱性の CVE 番号を検索すると、エクスプロイト情報が公開されている場合に具体的なスクリプトや手順の情報を参照できる。なお、このような公開情報は、脆弱性スキャンツールのレポート内容に含まれることが多い。

エクスプロイトを実際に行うツールとして、Metasploit や Core Impact Pro が挙げられる。これらのツールでは、実際に攻撃となる通信を行って脆弱性の有無を確認することができる。エクスプロイトの公開情報が無い場合やツールにエクスプロイト用モジュールが含まれていない場合には、脆弱性のエクスプロイトサービスを提供している検証サービス事業者への依頼も検討されたい。

以上の検証により、脆弱性スキャンで検出された脆弱性が実際に攻撃可能かどうかを確認できる。攻撃可能な脆弱性については、第三者に悪用されるリスクや影響度が比較的高いため、ソフトウェアの修正や緩和策の検討、ユーザへの注意喚起を早期に実施すべきである。

検証結果を踏まえて機器メーカが確認するポイント

- 検証対象機器に潜在する脆弱性に対するエクスプロイト情報を把握する
- 脆弱性エクスプロイトにより実際に脆弱性が攻撃可能かどうかを確認する
- 攻撃可能、かつ、深刻度の高い脆弱性については脆弱性への対応を早期に検討する

[15] EXPLOIT DATABASE https://www.exploit-db.com/

4.5.4 ハードコーディングパスワードの調査

（1） 検証の目的・趣旨

本検証の目的は、ハードコードされたパスワードを第三者が入手して不正アクセスにつながる可能性や、機器のコンポーネントにバックドアが仕掛けられていないかを確認することである。本検証では、バイナリデータに含まれるテキストデータを抽出し、パスワードに該当する文字列が無いかを確認する。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

ファームウェア解析ツール（Ghidra, IDA Pro 等）やバイナリエディタ（Stirling, GHEX 等）では、バイナリデータを 16 進数で表示する機能や ASCII コードや UTF-8 コード等の文字コードに該当するデータをテキストデータとして表示する機能がある。パスワードに該当する文字列が含まれる場合には、これらのテキストデータを確認してパスワードを特定可能である。目視により確認することも可能ではあるが、よく利用されるパスワードの辞書ファイルと照合することで既知のパスワードが含まれるかどうかを確認可能である。また、バイナリ解析の手法を活用して確認の対象を絞ることも可能である。例えば、認証におけるパスワードの照合ではパスワードの文字列と比較する処理が実行されるため、比較制御命令の引数となっている文字列データはパスワードに該当する可能性がある。

本検証により検証対象機器のファームウェアやバイナリコードに含まれるハードコードされたパスワードを確認可能である。OEM 製品や SDK を利用している場合、デバッグ用のバックドアとしてこのようなパスワードが残されたままになっているケースも想定される。機器メーカは、検出されたパスワードが実際に利用可能かどうかの検証も併せて検証サービス事業者に依頼することで、検出されたパスワードが悪用されるリスクについても把握可能である。

検証結果を踏まえて機器メーカが確認するポイント

- ファームウェアやバイナリコードに含まれるパスワードを把握する
- 検出されたパスワードが実際に利用可能かどうかを確認する

4.5.5 パスワードファイルの検証

（1） 検証の目的・趣旨

Linux や BSD 系をベースに開発されたシステムでは、/etc/shadow や/etc/passwd といったファイルにユーザ ID やハッシュ化されたパスワード等のアカウント情報が格納されている。攻撃者はこれらのアカウント情報を解析し、UART インタフェース経由のシェルコンソールへのログインする等に悪用する。本検証の目的は、ssh や telnet 等の実行ユーザのパスワードが第三者によって特定されうるかどうかを確認することである。本検証の結果として、ログインユーザのパスワードが報告される。このパスワードは検証対象機器へのログインで悪用される可能性があるため、OEM 製品や機器コンポーネントにこのような不備が含まれているかを確認することができる。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

本検証では、ファームウェアイメージから抽出したパスワードファイルに対応して検査を実施する。代表的なツールとして、John the ripper や Rainbow crack などがある。検証手法として、用意した辞書ファイルに含まれるパスワードを照合していく辞書攻撃による手法と、総当たりでパスワードの試行するブルートフォース攻撃による手法がある。辞書攻撃で利用する辞書ファイルについては、検証ツールに付属している辞書や Web 上で公開されている頻出パスワードリストが利用可能である。

検証サービス事業者が本手法により必ずしもパスワードを特定できるわけではない。また、どの程度まで辞書やブルートフォースの対象として検査するかによって検証時間が左右される。機器メーカは、本検証の依頼時のポイントとして、パスワードの強度としてどの程度のセキュリティレベルを要求するか、どこまで時間をかけて検証を実施するかを検証サービス事業者と協議することが望ましい。例えば、初期パスワードとして簡易なパスワードが設定されていないかを確認するものであれば、よく利用されるパスワードの辞書を用いた検査や短い文字数のブルートフォース攻撃による検査で十分なケースがある。

検証結果を踏まえて機器メーカが確認するポイント

- パスワードファイルに含まれるパスワードを把握する
- 検出されたパスワードが実際に利用可能かどうかを確認する

4.5.6 ネットワークサービスのパスワード検証

（1） 検証の目的・趣旨

第 4.5.5 項の手法では、ファームウェアから抽出したパスワードファイル等を対象にパスワード検証を行う。本検証は、検証対象機器が提供しているネットワークサービスポートで認証がかかっている場合に、ネットワークサービスのプロトコルに合わせたパスワード検証を行う。本検証においても第 4.5.5 項の手法と同様に初期パスワードで安易なパスワードが設定されていないか、第三者が特定可能かどうかを検証することを目的とする。また、ログイン試行回数をしきい値とした認証制限機能等のセキュリティ機能がある場合、本機能の有効性の検証も可能である。本検証を行うためのツールとして、THC Hydra や ncrack がある。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

本検証は次の流れで実施する。

1. 検証対象のプロトコルや認証機能の攻撃ポイントを確認する
2. 検証対象機器と検証用機器を互いに通信可能なネットワークに接続する
3. 検査ツールを用いて検査を実施する

まず、検証対象機器の通信に利用されているプロトコル情報や、認証が発生する攻撃ポイントに関する情報を収集する。後者について、SSH や Telnet のようにプロトコル自体に認証機能が備わっているものであれば検証ツールが自動で攻撃ポイントを設定するが、Web 管理機能の HTML フォームによるログイン画面の場合は、ID とパスワードの入力先となるパラメータの情報が検証時に必要となる。また、認証

の成功・失敗がプロトコルで定義されておらず、アプリケーションの挙動によって判断する場合には、認証が成功・失敗した場合のパターンの設定も同様に必要となる。

本検証により試行する ID やパスワードについては、よく利用される ID やパスワードを収録した辞書や総当たりで生成するデータ等を利用する。また、第 4.5.4 項や第 4.5.5 項の手法で検出されたパスワードを利用することも可能であるため、機器メーカは同手法とあわせて検証サービス事業者に依頼することで、効果的な検証を実施することができる。

検証結果を踏まえて機器メーカが確認するポイント

- ネットワークサービスへのログインに利用可能なパスワードを把握する
- ログイン試行回数の制限機能やログ機能等のセキュリティ機能の有効性を検証する

4.6 ファジング

ファジングは検証対象機器に対して不正又は想定外な入力を与え、適切に処理されるかどうかを確認するテスト手法である。第 4.5 節の検証手法では既知の脆弱性があるかを検査することが目的であったのに対し、ファジングは未知の脆弱性を検出することを目的とする手法となっている。ファジングにより検出される脆弱性の例として、バッファオーバーフロー脆弱性や例外処理の不備がある。旧来のファジング手法は、ランダムにテストデータを生成して異常な動作をしないかを確認するのみであったが、近年では、プロトコルを基準にテストデータを生成する手法やコード網羅率を指標として、効率的に検査する手法が実用化されている。本節では、ファジングの代表的な手法であるファイルフォーマット・ネットワークプロトコルをベースとする手法とコード網羅率を指標とする手法について紹介する。

4.6.1 ファイルフォーマット・ネットワークプロトコルベースのファジング手法

（1） 検証の目的・趣旨

本検証では、ファイルフォーマットやネットワークプロトコルに基づくファジングにより、未知の脆弱性を検出することを目的とする。ファジングでは、不具合を起こしうる異常なデータをテストデータとして利用する。ただし、一般的なプログラムでは入力データが与えられたときの入力チェックを行っており、ファイルフォーマットやネットワークプロトコルに準拠しないデータは異常又は例外データとして扱われる。このとき、入力チェックの段階で多数のテストデータが弾かれるため、テスト効率が悪いものとなってしまう。そのため、ファイルフォーマットやネットワークプロトコルのデータ形式に沿った形のテストデータを生成する手法がある。本手法では入力チェック後の機能や処理のテストにつながり、ファジングにおける不具合の検出可能性を高めることができる。本検証を行うためのツールとして、Peach Fuzzer や beSTORM 等のファジングツールがある。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

ファイルフォーマットやネットワークプロトコルをベースとするファジング手法による検証は、主に次の流れで実施する。

1. 検証対象のファイルフォーマット・ネットワークプロトコルの選定

2. 検証対象の攻撃ポイントの選定
3. バイナリコード実行環境やネットワークの構築
4. ファジング実行
5. ファジング結果の分析

まず検証対象となるファイルフォーマットやネットワークプロトコルを選定する。検証対象機器のバイナリコードに対するファジングの場合、入力データとして与えるファイルに応じたフォーマットを選定し、ネットワークサービスに対するファジングの場合は通信プロトコルに合わせたネットワークプロトコルを選定する。ファジングツールによって対応するフォーマットやプロトコルが異なるため、検証対象に合わせてファジングツールの選定を行う。次にファジング用に生成したテストデータを入力するための攻撃ポイントを選定する。攻撃ポイントの例として、ファイルの入力処理やネットワーク通信のサービスポートが該当する。ファジングの実施には、ファジングツールからこれらの攻撃ポイントへテストデータを入力できる環境も併せて必要である。

本検証の結果として、脆弱性や不具合を引き起こすテストデータが得られる。これらのテストデータによる不具合発生時の設定や状態の確認、バイナリコード内の該当箇所の特定についてもあわせて依頼することでソフトウェアの修正をスムーズに行うことができる。

検証結果を踏まえて機器メーカが確認するポイント

- 検証で得られたテストデータを基に不具合の原因を確認する
- 不具合が脆弱性につながる内容の場合、脆弱性に対するソフトウェアの修正のほか、緩和策について検討する

4.6.2 コード網羅率を指標とするファジング手法

（1） 検証の目的・趣旨

代表的なファジングツールでは、コード網羅率を評価指標とした遺伝的アルゴリズムを応用することで、より効果的なファジングを可能にしている。コード網羅率は対象のバイナリコードに含まれる処理がどの程度実行されたかを表す指標であり、ファジングにおいてはコード網羅率が向上するテストデータを優先的に利用して新しいテストデータを生成することで、不具合を検出率向上につなげていく。本検証では、バッファオーバーフロー等の脆弱性や不具合の検出を目的として、コード網羅率を指標としたファジングによるテストを行う。コード網羅率を指標とする代表的なファジングツールとして、AFL や AFL++がある。これらのツールは、基本的にバイナリコード単体を対象としたテストであり、検証対象機器のシステム全体を動作させた状態でのテストについては、第 4.6.1 項の手法での検証が必要となるため注意されたい。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

本検証の主な流れは次のとおりである。

1. コード網羅率の観測方法の選定
2. 検証対象の攻撃ポイント等の選定
3. ファジング実行

4. ファジング結果の分析

ファジングにおけるコード網羅率の計測手法として、ファジングツールに付属する専用のコンパイラを用いてコンパイル時に計測用コードを注入する手法、バイナリコードの実行時に動的に計測用コードを注入する手法がある。前者の手法では、バイナリコードのソースコードが必要であり、検証サービス事業者へソースコードを提供するか、機器メーカ側でコンパイルしたバイナリコードを提供することとなる。OEM 製品や SDK 等でソースコードが入手できない場合や、検証サービス事業者へソースコードを提供できない場合は、後者の動的に計測用コードを注入する手法が利用可能である。

検証対象の攻撃ポイントとして、コマンドラインベースのバイナリコードに対するファイルの入力や標準入力を攻撃ポイントとして入力するツールが主流となっている。それ以外の攻撃ポイントとして、TCP 等のネットワーク通信を攻撃ポイントとするツールもある。

本検証の結果として、脆弱性や不具合を引き起こすテストデータが得られる。これらのテストデータによる不具合発生時の設定や状態の確認、バイナリコード内の該当箇所の特定についてもあわせて依頼することでソフトウェアの修正をスムーズに行うことができる。

検証結果を踏まえて機器メーカが確認するポイント

- 検証で得られたテストデータを基に不具合の原因を確認する
- 不具合が脆弱性につながる内容の場合、脆弱性の対応に係るソフトウェアの修正のほか、緩和策について検討する

4.7 ネットワークキャプチャ

ネットワークキャプチャでは、検証対象機器がネットワーク上で送受信するデータを精査し、セキュリティ上の課題が無いかを検証する。具体的には、ネットワーク上で流れる情報にパスワード等の機密性の高い情報が含まれているか、適切に暗号化されているか確認する。また、OEM 製品や SDK 等のサプライチェーン上で含まれたコンポーネントが提供する機能の通信において意図しない通信が行われる場合がある。例えば、OEM メーカや SDK メーカが提供するサーバを経由して外部アクセスする機能が存在し、プライバシーポリシーの問題につながった事例がある。本検証ではこのような意図しない通信の有無を確認することが可能である。

4.7.1 Ethernet ネットワークパケットキャプチャ

（1） 検証の目的・趣旨

本検証では、Ethernet 通信のパケットデータをキャプチャし、その内容を精査する。検証時には、これらのデータの中に機密性の高い情報や攻撃へのヒントとなる情報が含まれているかどうかを確認する。暗号化された経路での通信を想定している場合には、適切に暗号化されているかどうか、キャプチャしたデータを基に確認する。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

本検証では、検証対象機器の通信をキャプチャするための仕組みが必要となる。代表的な手法として、ポートミラーリング機能を有するネットワークスイッチを用いる手法がある。ポートミラーリング機能は、スイッチングハブ等における所定のポートの通信パケットを他のポートにも同様に送信する機能である。この機能を用いて、検証対象機器の通信パケットを別の検証用端末で受信して解析できる。検証用端末では、Wireshark や tcpdump 等のツールを用いて受信したパケットを解析する。

各検証ツールでは、パケットの内容を自動的に解析し、プロトコルに合わせた構造データを確認できる。構造化したデータを ASCII コードで表示する機能があり、パスワード等の情報が含まれていないかを確認できる。また、これらの内容から通信が暗号化されているかどうかを確認できる。

機器メーカは、本検証結果を基に、通信で流れるデータが想定しているものかどうか、通信が暗号化され情報が適切に保護されているかどうかを確認する。

検証結果を踏まえて機器メーカが確認するポイント

- 通信を暗号化している場合、適切に暗号化されているかどうかを確認する
- 検証対象機器の仕様やセキュリティポリシーにそぐわない機密性の高い情報が流れていないかどうかを確認する
- 検証対象機器の通信先のサーバやネットワークを確認する

4.7.2 Bluetooth パケットキャプチャ

（1） 検証の目的・趣旨

Bluetooth は主にデータの送受信や機器のコントロールに利用される。一般に、ペアリングのプロセスで暗号鍵の交換や認証を実施するため、特定の相手のみ通信可能となる。ただし、仕様によっては認証がされないまま利用されることや、Bluetooth プロトコル自体の脆弱性により第三者が Bluetooth 機能を介して機器の制御を乗っ取る可能性がある。本検証では、Bluetooth の通信内容をキャプチャすることで Bluetooth アプリケーションレベルのプロトコルや通信で流れる情報を確認し、通信における情報漏えいや不正アクセスのリスク、セキュリティ上の課題等を検証する。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

本検証の主な手法として、検証対象機器と通信するアプリケーションを検証用機器上で動作させ、アプリケーションと検証対象機器の Bluetooth 通信をキャプチャする手法がある。キャプチャ用のツールには、Bluetooth 用ライブラリ BlueZ に付属しているツール hcidump や Android OS の HCI スヌープログ機能等がある。キャプチャしたデータは pcap フォーマット[16]で出力し、パケットキャプチャツール Wireshark を利用して内容を確認することができる。この流れで検証対象機器の Bluetooth 通信をキャプチャし、利用されている Bluetooth プロファイルやペイロードデータを解析することが可能である。

[16] パケットキャプチャライブラリ libpcap, Npcap のパケットキャプチャファイル用フォーマット。

OEM 製品や SDK を利用している機器の場合、機器メーカが意図しないパスワードや個人情報等の機密性の高いデータが含まれる可能性がある。本検証により機密性の高い情報が含まれているかどうかを確認可能である。また、Bluetooth の通信内容から検証対象機器の制御命令やリプレイ攻撃の可能性が確認できた場合には、攻撃者が検証対象機器とペアリングした際の不正な制御につながるリスクも想定されるため注意が必要である。

検証結果を踏まえて機器メーカが確認するポイント

- Bluetooth 通信で送受信される情報や制御データ、プロファイル等を把握する
- 検証対象機器の仕様やセキュリティポリシーにそぐわない機密性の高い情報が流れていないかどうかを確認する

4.7.3 アプリケーション通信のキャプチャ

（1） 検証の目的・趣旨

本検証は、第 4.7.1 項の手法よりも上位のアプリケーションレベルのネットワーク通信のキャプチャにより送受信される情報を確認する。各アプリケーションのプロトコルに特化したツールを利用することで、検証作業の効率化につながり、より詳細な情報が得られることがある。また、本検証ではプロキシサーバの仕組みを利用して通信をキャプチャする手法があり、TLS のような公開鍵暗号方式による暗号通信が行われる場合におけるデジタル証明書の検証不備等も検出可能である。

（2） 検証の流れと検証結果を踏まえた機器メーカの確認ポイント

本検証を実施する際のツールとして、Burp Suite や fiddler、mitmproxy 等がある。ネットワーク通信をキャプチャする上では Web ブラウザやモバイルアプリケーションを操作し、実際に発生した内容をキャプチャする必要がある。先述のツールでは、Web 管理画面からリンクされているページを自動的に洗い出し、作業の自動化することも可能である。HTTP 及び Web 機能の検証の主な流れは次のとおりである。

1. 検証用ツールを起動し、プロキシサーバを動作させる
2. 検証用機器の Web ブラウザ等でプロキシサーバを設定する
3. 検証対象機器の Web 管理画面等にアクセスし、通信を発生・キャプチャする
4. キャプチャした通信内容を確認する

本検証実施時の注意点として、暗号化通信におけるデジタル証明書に関する内容がある。本検証ではアプリケーションレベルのプロキシサーバを経由した通信となるため、暗号化された通信の場合、サーバ証明書の不整合による通信エラーが発生する。そのため、各ツールで独自に作成される認証局のサーバ証明書を検証用機器に事前にセットアップする必要がある。これにより、サーバ証明書関連のエラーを回避して、検証を実施することが可能である。

本検証結果を踏まえ、これまでのネットワークキャプチャに関する検証手法と同様に、通信内容に機密性の高い情報が送受信されていないかどうか、送受信されている場合には適切に保護されているかどうかを確認する。また、サーバ証明書関連のセットアップを行わない状態で暗号化通信が成立する場合、アプリケーション側のサーバ証明書の検証不備の可能性がある。この場合、中間者攻撃による通信データの

盗聴やデータ改ざんの可能性があるため対策が必要である。また、OEM 製品や SDK を利用した製品の場合、意図しない機能やページ、バックドアにつながるような内容が無いかどうかを検証結果から確認する。

検証結果を踏まえて機器メーカが確認するポイント

- 通信を暗号化している場合、適切に暗号化されているかどうかを確認する
- 検証対象機器の仕様やセキュリティポリシーにそぐわない機密性の高い情報が流れていないかどうかを確認する
- 検証対象機器の通信先のサーバやネットワークを確認する

4.8 検証依頼時に用意することが望まれる情報

機器メーカが検証を依頼する際に、検証対象となる機器のほか、当該機器に関する情報やデータを提供することで、検証サービス事業者は検証にかかる時間を短縮しつつ正確な検証を実施し、より効果的な検証につなげることができる。本節では、検証をより効果的にするために機器メーカが用意すべき情報やデータについて紹介する。実際の内容については、検証を依頼する際に検証サービス事業者と協議し、依頼する内容や契約等の状況に応じて決定することが必要である。

検証依頼時に用意することが望まれる情報を表 4-3 に示す。攻撃者が検証対象機器を解析することを想定し、ブラックボックスでの検証を依頼する場合、機器メーカはインターネットや取扱説明書等での公開情報を検証サービス事業者へ提供することが望まれる。検証にかかる工程や時間短縮を図るために、ホワイトボックスでの検証を依頼する場合、機器メーカは検証サービス事業者に対してできるだけ多くの情報を提供することが望ましい。例えば、ソフトウェアのアーキテクチャに関する情報は、バイナリ解析により特定しうる情報ではあるものの、これらの情報を機器メーカが検証サービス事業者に提供することで解析にかかる時間短縮やコスト低減につなげることが可能である。一方、用意できない情報がある場合には、検証コストが高くなるケースや検証自体が難しくなるケースもある。そのため、機器メーカは検証サービス事業者へ用意すべき情報を検証依頼時に確認する必要がある。

表 4-3 検証依頼時に用意することが望まれる情報

項	検証手法	用意することが望まれる情報
4.2.1	ハードウェアの情報収集	• チップのデータシート（メーカ、型番、基板上の配置等） • OS、BIOS、ブートローダ、ミドルウェアに関する情報
4.2.2	デバッグインタフェースの特定	• デバッグインタフェース情報（種別、ピン配置） • OS、BIOS、ブートローダ、ミドルウェアに関する情報
4.2.3	デバッグインタフェースからのファームウェア抽出	• デバッグインタフェース情報（ボーレート、パリティ等） • メモリアドレス情報（RAM、フラッシュメモリ等） • OS、BIOS、ブートローダ、ミドルウェアに関する情報
4.2.4	ファームウェア解析	• ファームウェアイメージファイル

項	検証手法	用意することが望まれる情報
		• OS、BIOS、ブートローダ、ミドルウェアに関する情報 • ファームウェアのアーキテクチャ（ファイルシステム、暗号化アルゴリズム等）
4.3.1	バイナリ解析とバッファオーバーフロー脆弱性の特定	• バイナリコード • バイナリコードのシンボル情報
4.3.2	シンボリック実行を利用したバイナリ解析	• バイナリコード • バイナリコードのシンボル情報
4.3.3	APK ファイルのデコンパイル	• APK ファイル
4.4.1	ネットワークポートスキャン	• ネットワークサービス設計書（ネットワークインタフェース、サーバソフトウェア、バージョン、プロトコル、待ち受けポート等）
4.5.1	ソフトウェア BOM の構築・脆弱性スキャン	• ファームウェアイメージファイル
4.5.2	ネットワーク経由の脆弱性スキャン	• ネットワークサービス設計書（ネットワークインタフェース、サーバソフトウェア、バージョン、プロトコル、待ち受けポート等） • ネットワークサービスの認証情報
4.5.3	脆弱性エクスプロイト	• ソフトウェア設計書 • ネットワークサービス設計書（サーバソフトウェア、バージョン、プロトコル、待ち受けポート等） • ネットワークサービスの認証情報
4.5.4	ハードコーディングパスワードの調査	• ファームウェアイメージファイル
4.5.5	パスワードファイルの検証	• パスワードファイル • サービスの認証情報
4.5.6	ネットワークサービスのパスワード検証	• サービスの認証情報
4.6.1	ファイルフォーマット・ネットワークプロトコルベースのファジング手法	• ネットワークサービス情報（サーバソフトウェア、バージョン、プロトコル、待ち受けポート等） • バイナリコード • バイナリコードのシンボル情報 • デバッグインタフェース情報（ボーレート、パリティ等）
4.6.2	コード網羅率を指標と	• ネットワークサービス情報（サーバソフトウェア、バージョン、

項	検証手法	用意することが望まれる情報
	するファジング手法	プロトコル、待ち受けポート等） • バイナリコード • バイナリコードのシンボル情報 • デバッグインタフェース情報（ボーレート、パリティ等）
4.7.1	Ethernet ネットワークパケットキャプチャ	• ネットワークサービス設計書（ネットワークインタフェース、サーバソフトウェア、バージョン、プロトコル、待ち受けポート等）
4.7.2	Bluetooth パケットキャプチャ	• Bluetooth 接続情報（Bluetooth バージョン、認証方式等） • Bluetooth アプリケーションの通信情報（プロファイル、プロトコル等）
4.7.3	アプリケーション通信のパケットキャプチャ	• ネットワークサービス設計書（ネットワークインタフェース、サーバソフトウェア、バージョン、プロトコル、待ち受けポート等）

5 検証結果を踏まえた対応の検討

本章では、4 章のセキュリティ診断によって得られた結果に対する対応のポイントについて紹介する。それぞれの対策は診断結果を基にリスク評価を実施し、リスクレベルやコストを考慮しながら対策の方針を決定する。

5.1 検証報告に対する機器メーカの対応

機器メーカは、検証サービス事業者より検証結果を受領した後、セキュリティ課題への対応方針を定める必要がある。この際、対応した結果を記録し、管理することが望まれる。この対応記録は、今後の開発において有益な資産として活用できるため、問題を作り込んだ原因や原因箇所、解決方法などをセキュリティ課題ごとに整理すると良い。セキュリティ課題に対する対応記録の例を表 5-1 に示す。この例では「検証サービス事業者の報告事項」と「機器メーカ記載事項/処置内容」に分けて、双方で対応しておくべき記載内容の例を提示している。

表 5-1 セキュリティ課題に対する対応記録の例 [17]

項目名	報告者	記載内容の説明
1）検証サービス事業者の報告事項		
管理番号	検証サービス事業者	課題管理番号 ※ プロジェクトを通じて一意の値が望ましい。また、同一原因の管理番号を記載できることが推奨される。
プロジェクト名	検証サービス事業者	対象製品のプロジェクト名
深刻度基準	検証サービス事業者	CVSSv3 を基準として深刻度を記載
起票日	検証サービス事業者	レポートの報告日を記載
レポートタイトル	検証サービス事業者	レポートのタイトル ※ 200 文字以内で、簡潔かつインシデントの重要性を示すタイトルが望ましい。
発行者名	検証サービス事業者	レポートの起票者名を記載
内容	検証サービス事業者	下記の記載内容をカバーすること ・試験環境 ・設定条件 ・試験手順 ・発生する問題 ・期待する動作

[17] CCDS『IoT セキュリティ評価検証ガイドライン Rev1.0』より加筆、修正
https://www.ccds.or.jp/public_document/index.html#Verification_guidelines1.0

項目名	報告者	記載内容の説明
		・発生条件
添付ファイル	検証サービス事業者	内容を補足するログデータ等、添付資料を付加する。
機能名	検証サービス事業者	対象のシステムやソフトウェアの機能名を記載する
検出バージョン	検証サービス事業者	課題を確認したバージョンを記載する
発生環境	検証サービス事業者	課題を確認した環境を記載する 例）テスト環境など
発生頻度	検証サービス事業者	試行回数に対する現象の発生回数を記載する 例）XX（発生回数）/YY（試行回数）
発見工程	検証サービス事業者	発見した工程
発見手段	検証サービス事業者	コードレビューやテスト名称等、現象を発見した工程名称を記載する。
検証項目番号	検証サービス事業者	課題を検出した評価検証項目の番号を記載する。
検証シナリオ名	検証サービス事業者	課題を検出した検証シナリオ名（あるいはシナリオ番号）を記載する。
影響を受けるシステム	検証サービス事業者	課題によって影響が想定される機能名やサブプロセスを記載する。
想定される影響	検証サービス事業者	課題によって、どのような影響が想定されるのか具体的な内容を記載する。
参考情報	検証サービス事業者	JVN、NVD における関連インシデント番号。CVE 番号や CWE 識別子等、解析に役立つ情報や関連情報を記載する。
課題是正対策の案	検証サービス事業者	検出課題について、対応すべき対策案を提示する
２）機器メーカ記載事項/処置内容		
ステータス	機器メーカ	レポートのステータス 例）課題確認中、改修中、改修完了、クローズ等
クローズ理由	機器メーカ	レポートをクローズした理由を記載 例）改修済、次機種検討、現状通り、ドキュメント修正
処置担当者	機器メーカ	現象の原因調査や改修対応等の処置に対する担当者名を記載する。
処置完了日	機器メーカ	処置の完了日を記載する。
発生原因	機器メーカ	課題の発生原因を記載する。
原因箇所	機器メーカ	課題が発生した原因を含むソフトウェア成果物。仕様書であれば仕様書名（ファイル名）とそのページ、行数など。ソースコードであれば、ファイル名、関数名、行数なし。
対応見積工数	機器メーカ	修正を行う場合の見積工数を記載する

項目名	報告者	記載内容の説明
解決方法/処置内容	機器メーカ	解決方法、修正内容あるいは対応方針を記載する。
処置区分	機器メーカ	機器メーカとしての対応方針を記載する。 例）仕様通り、プログラム改修、ドキュメント対応、次機種対応、対応見送り等。
修正対象	機器メーカ	修正や改修を行った仕様書名や、ソフトウェアコードのファイル名等を記載する。
修正バージョン	機器メーカ	修正を行ったバージョン名称を記載する。
リリース日	機器メーカ	修正バージョンのリリースに日時を記載する。
3）その他、分析やフォードバックに活用する項目		
不具合区分	機器メーカ	課題の原因となった不具合の分類区分を記載する。今後のフィードバックや分析に活用できるよう IPA 発行の ESBR[18]等に準拠した分類項目が望ましい。
作り込み工程	機器メーカ	課題の原因となる不具合を作り込んだ工程を記載する。 例）システム要求定義（システム要求分析）、システムアーキテクチャ設計（システム方式設計）、ソフトウェア要求定義（ソフトウェア要求分析）、ソフトウェアアーキテクチャ設計（ソフトウェア方式設計）、ソフトウェア詳細設計（ソフトウェア詳細設計）、実装（コーディング）。
調査工数	機器メーカ	課題の原因調査に要した工数を記載する。
処置工数	機器メーカ	課題の改修対応に要した工数を記載する。
発見すべき工程	機器メーカ	本来課題や原因となった不具合を発見すべき工程を記載する。
発見すべきアクティビティ	機器メーカ	本来課題や原因となった不具合を発見すべきアクティビティを記載する。 ※アクティビティとは工程作業をさらに分割し、順序付けした作業要素を指す。

[18] IPA 『組込みソフトウェア開発における品質向上の勧め[バグ管理手法編]』
https://www.ipa.go.jp/sec/softwareengineering/std/emb.html

5.2 検証結果に基づくリスク評価と対応方針の検討

5.2.1 検出課題に対するリスク評価の実施

検証事業者によって検査を実施した結果、検出された課題については、リスク評価を行い、対応優先度や対策方針を検討する上での指標とする。リスク評価の手法については、多くの手法、基準が定義されており、対象製品あるいは企業側が求めるニーズに応じて、選定する必要がある。国内外の代表的なリスク評価手法を表 5-2 に示す。

表 5-2 国内外のリスク評価手法 [19]

該当文書・リスク評価手法	発行機関	特徴
『IoT セキュリティ・セーフティ・フレームワーク』 3-2 フィジカル・サイバー間をつなげる機器・システムに潜むリスクの整理	経済産業省	「回復性困難度の評価」の軸と「経済的影響の評価」の軸によって機器やシステムのリスクを 9 象限にて分類する。
CVSSv3	FIRST（Forum of Incident Response and Security Teams）	脆弱性が「機密性」、「完全性」、「可用性」に対する影響を主軸に評価を行う。
CVSSv3 拡張版 『CCDS 製品分野別セキュリティガイドライン』	CCDS スマートホーム WG	CVSSv3 を基に、「生命・財産への影響」、「情報の重要度」をリスクファクターとして追加。
ETSI TS102 165-1	欧州電気通信標準化機構（ETSI）	「攻撃に要する時間」や「資産への影響」など、CVSS にはないパラメータが定義されている。
OCTAVE Allegro	カーネギーメロン大学（米国）	組織によるセキュリティニーズを評価することを目的とした手法。「安全・健康」がパラメータに組み込まれている。
The OWASP Risk Rating Methodology	OWASP（Open Web Application Security Project）	「脅威の要因」、「脆弱性の要因」、「テクニカルインパクト」、「ビジネス上の影響」の平均値によってリスクを評価する。
FAIR	Risk Management Insight LLC	「損失発生度（LEF）」、「金銭的な影響度（PLM）」によってリスクを評価する。

[19] CCDS 『IoT セキュリティ評価検証ガイドライン 第 1 版』より加筆、修正
https://www.ccds.or.jp/public_document/index.html#Verification_guidelines1.0

表 5-2 に挙げたリスク評価手法より、機器メーカとして活用しやすい手法として、「IoT セキュリティ・セーフティ・フレームワーク」で定義されたリスク評価手法及び CVSSv3（CVSSv3 拡張版）を使用した実施事例を以下に示す。

まず「IoT セキュリティ・セーフティ・フレームワーク」のリスク評価手法は、図 5-1 に示すように、インシデントの影響について、回復の困難性を示す度合いの判断軸（横軸）と経済的影響の度合いの判断軸（縦軸）を組み合わせて、9 つの象限（カテゴリ）に対象機器の潜在的なリスクをマッピングすることができる。この 9 象限を用いた分析の結果、右上にカテゴライズされればインシデントの影響が大きい傾向にあり、より厳格な対策が必要となる。他方で、左下にカテゴライズされるほど、軽微な対策で対応が可能となる。図 5-2 には、この 9 象限に家庭用のネットワークカメラをマッピングした結果を示す。回復の困難性の判断にあたっては、撮影映像、録画映像などのプライバシー情報の漏えいが、精神的にも回復が影響を与える可能性もあるため、「重大なダメージ」として定義できる。また経済的影響の判断では、インシデント発生時の影響は、個人への賠償に留まるものであるため「限定的な経済的な経済影響」として定義できる。このリスク評価手法では、上記のように比較的簡易な分析で、機器やシステムの潜在的なリスクを検討できるため、リスクの対応方針の初期分析においては有用である。

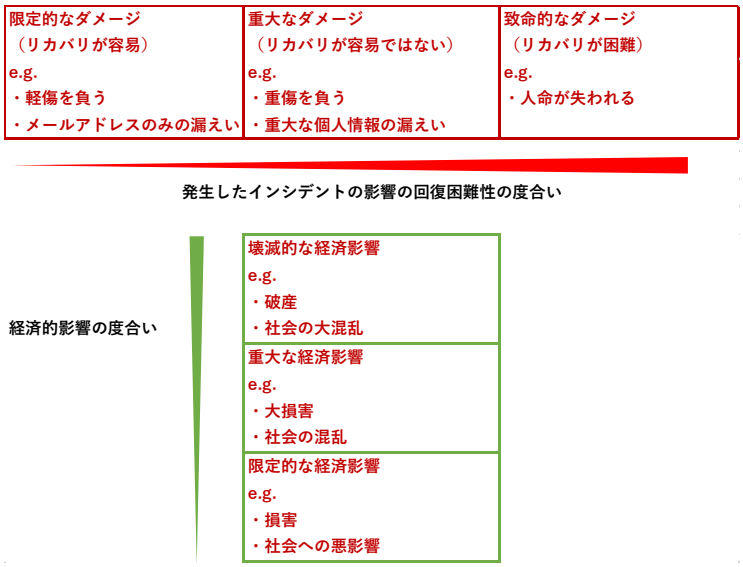

図 5-1 発生したインシデントの影響の回復困難性と経済的影響の度合いのイメージ [20]

[20] 経済産業省 「IoT セキュリティ・セーフティ・フレームワーク」
https://www.meti.go.jp/press/2020/11/20201105003/20201105003.html

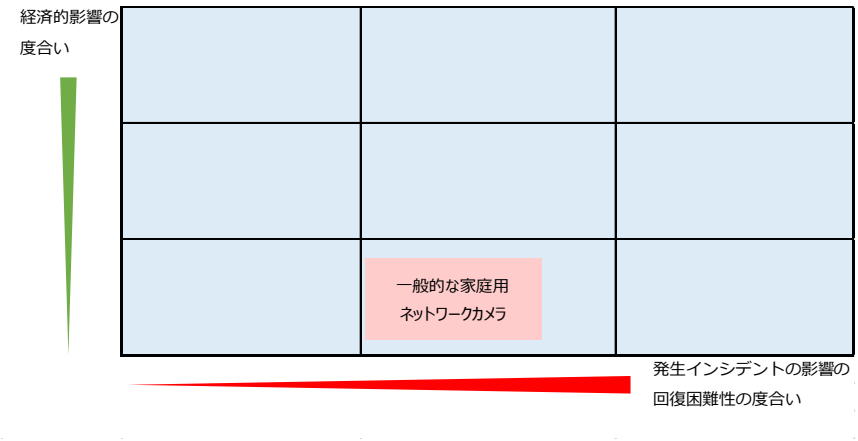

図 5-2 家庭用ネットワークカメラのリスク評価事例（9 象限へのマッピング）

次に CVSSv3 における実施事例を示す。表 5-3 に示すとおり、CVSS は検証の結果検出された課題について、深刻度・対応緊急度を定量的に判断することが可能であり、改修優先度の判断指標としても活用できる。

表 5-3 CVSSv3 における深刻度のスコア表

CVSS スコア	深刻度
9.0～10.0	緊急（Critical）
7.0～8.9	重要（High）
4.0～6.9	警告（Medium）
0.1～3.9	注意（Low）
0.0	なし

CVSS は図 5-3 のように「基本評価値」、「現状評価値」、「環境評価値」の三つの評価値から構成される。基本評価値は、脆弱性そのものの特性を示す値であり、主にセキュリティ課題を検出した検証技術者が評価を行う。現状評価値は、攻撃コードや対策情報の公開有無などの補足情報を加えて、補正した値となり、主に開発者が評価を行う。環境評価値は対象機器の実際の利用環境や、セキュリティ対策状況を踏まえて、再計算する値であり、主に運用関係者が評価を行う。このように CVSSv3 は脆弱性そのものの深刻度に対して、異なる視点から補正や再計算を行い、より現実に即した値を算出することが可能であり、課題対策の優先度を決定する上で、有用な指標となる。CVSSv3 の具体的な計算式については、IPA の公開情報 [21]等を参照されたい。

[21] IPA 共通脆弱性評価システム CVSS 概説
https://www.ipa.go.jp/security/vuln/CVSS.html

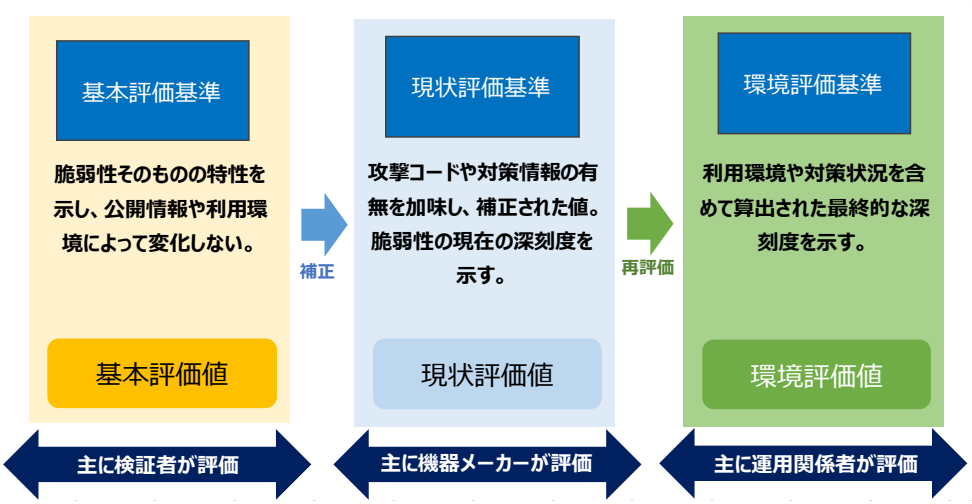

図 5-3 CVSSv3 における三つの評価軸

一方で、CVSS では脆弱性が機密性、完全性、可用性に与える影響を主軸とするリスクを評価する手法であるため、資産損失額や人命、健康に与える影響などは、リスクファクターとして加味されていない。CCDS は、この点を踏まえて CVSSv3 の拡張版 [22]をスマートホームのガイドラインの中で定義している。スマートホームにおいては、個人情報などの機密性が高い情報や、人命や利用者の資産に影響するリスクも想定されるため、「情報の重要度（II）」、「生命・財産への影響（LP）」というリスクファクターを従来の CVSS に加えた定義を提唱している。表 5-4 に CVSSv3 を使用したリスク評価の実施例を示す。なお、脆弱性の事例については、実際に脆弱性データベースに登録されている報告内容を基に記載している。

この例より、基本評価値として算出された値が、現状評価値ではやや低い値に補正されていることが確認できる。また、環境評価値では、脆弱性への対策により機密性、完全性、可用性に対するリスク対処が行われたため、注意レベルにまでリスクが低減されていることが分かる。この実施事例が示すように、CVSSv3 を用いたリスク評価では、セキュリティ対策を実施した結果のリスクの低減についても、定量的に算定することができる。そのため、実施すべきセキュリティ対策の優先度設定や対費用効果を把握する上での指標としても有用である。

表 5-4 ネットワークカメラの脆弱性に対する CVSSv3 を使用したリスク評価

脅威分類	脆弱性の事例	基本評価値	基本評価値ランク	現状評価値	現状評価値ランク	環境評価値	環境評価値ランク
データ改ざん	ネットワークカメラにおける OS コマンドインジェクションの脆弱性	8.8	重要	8.2	重要	2.7	注意

[22] CCDS 『製品分野別セキュリティガイドライン_スマートホーム編 1.0 版』
https://www.ccds.or.jp/public_document/index.html#guidelines-smarthome_1.0

脅威分類	脆弱性の事例	基本評価値	基本評価値ランク	現状評価値	現状評価値ランク	環境評価値	環境評価値ランク
情報漏えい	ネットワークカメラのファームウェアにおけるアクセス制御に関する脆弱性	7.2	重要	6.7	警告	1.2	注意

5.2.2 検出課題を含むリスク対応方針の検討

次のステップでは、リスク評価の実施結果を踏まえて、検出課題や想定されるリスクへの対応方針の検討を行う。想定されるリスクとしては、検証によって検出された課題のような対象機器側で対策可能なものだけではなく、オペレータなど関係者による人為的なミスや内部犯行も含まれる。加えて、事故や災害時による障害なども含まれるため、対象機器のユースケースや運用、廃棄といったライフサイクル全体を見据えた検討が必要となる。製品のライフサイクルや次期モデルの検討において、対策すべきタイミングを以下に示す。

- **製品（対象機器）の出荷前に対策すべき課題やリスク**
  - 検出課題のうち、リスク評価において深刻度/重要度が高いもの
    （例.CVSSv3 基準において、7.0 以上など）
  - 想定される発生頻度が高く、機密性の高い資産や、重要度の高い資産に影響するリスク
- **製品（対象機器）の出荷後に対応するべき課題やリスク**
  - 検出課題のうち、深刻度が比較的軽微なもの
  - 検出課題のうち、ソフトウェアアップデートや、運用による回避、低減が可能なもの
- **製品の出荷後、運用フェーズにおいて継続的に対応が必要な課題やリスク**
  - 認証情報や証明書の更新など、継続的な対応を要するもの
  - オペレータや関係者に人為的なミスや内部犯行によって発生するもの
- **次期モデルの検討において対応すべき課題**
  - 検出課題のうち、仕様や設計（アーキテクチャ）に原因があり、現行機種では対応が困難なもの

表 5-5 及び図 5-4 に情報セキュリティにおけるリスク対応の考え方を示す。想定されるリスクや課題に対しては、リスク回避あるいはリスク低減対策を行うことが望ましいが、課題やリスクの内容によっては、対策が困難なケースや、対費用効果を考えると現実的とはいえない場合もある。そうしたケースについては、リスク評価結果を基に、リスク保有やリスク移転の判断を行うことも必要となる。いずれのケースであっても課題やリスクへの対応方針決定は、企業全体の資産損失に影響する問題であり、経営層を含めた協議によって、意思決定を行うことが望ましい。

表 5-5 情報セキュリティにおけるリスク対応の考え方

手法		説明
リスク回避		リスク発生の根本的要因（作業、事象等）を排除することにより、リスクを回避する
リスク低減	損失予防	損失の発生頻度を減少させることによってリスクを低減する
	損失軽減	損失の度合いを小さくすることによって、リスクを低減する
	リスク分離	資産や資源を分化、分散することによってリスクを低減する
	リスク集中（結合）	損失を受ける資産や資源を集中化することによってリスクを低減する
リスク保有		リスクが顕在化したときに発生する損失を組織自体の財務力で負担する
リスク移転		契約等を通じてリスクを第三者へ移転することによってリスクを低減する（損失負担のリスクを外部へ転嫁する）

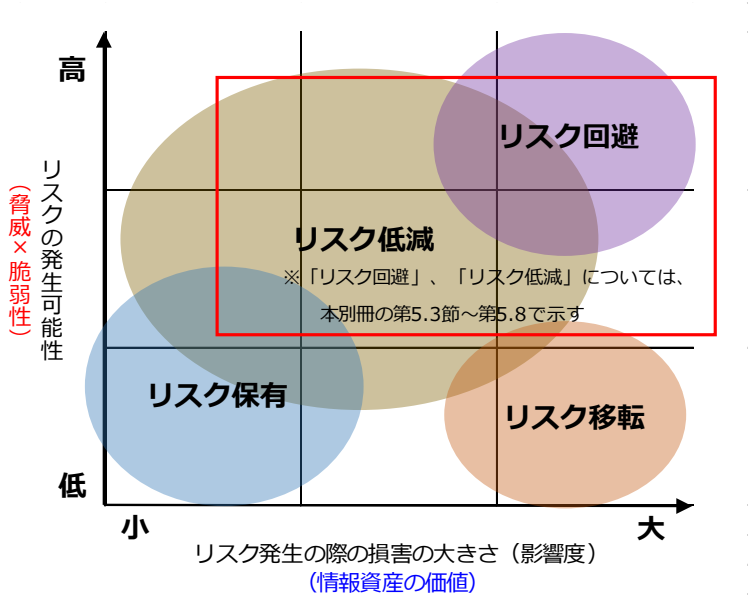

図 5-4 リスク対応に関する概念図 [23]

以降では、本別冊の第 4 章で記載した検証手法によって検出されたセキュリティ課題への「リスク回避」、「リスク低減」の方法について記載する。

[23] IPA 情報セキュリティマネジメントと PDCA サイクルに基づき加筆
https://www.ipa.go.jp/security/manager/protect/pdca/risk.html

5.3 ハードウェアハッキング・ファームウェア解析の結果を踏まえての対応

ハードウェアハッキングでは、ハードウェアから収集可能な情報やデータにアクセス・抽出可能かどうかを検証する。また、ファームウェア解析では、ファームウェアに含まれる情報やデータを解析する。検証で得られる情報は脆弱性の特定や機密情報の収集につながる可能性があるため、機器メーカは検証対象機器についてハードウェアやファームウェアから収集可能な情報を把握し、対策を講じることが望ましい。これらの代表的な対応手法を以下に示す。

- **不要なデバッグインタフェースを無効化する**：不要なデバッグインタフェースを利用できることが判明した場合、UART や JTAG 等のデバッグインタフェースを無効化し、RAM へのアクセスやファームウェア抽出を防止する。必要な機能として有効化している場合においても、第三者が収集可能な情報を少なくするため、コンソール機能を無効化、ログの出力を最小限に設定することが望ましい。例えば、U-Boot や Linux kernel ではシリアルコンソールのシェル機能やログ出力を無効化する設定が可能である。
- **ファイルシステムを暗号化する**：機器に対するセキュリティ要件等でファームウェア解析への対応が求められる場合、ファームウェア内のファイルシステムを暗号化し、情報の解析を難しくする。ただし、復号用の鍵データが耐タンパ性 [24]のあるハードウェアに格納されていない場合は、第三者により暗号アルゴリズムや鍵データを解析され、暗号化されたファームウェアが復号される可能性がある。このような暗号化では情報収集を難しくする緩和策であることを注意されたい。

5.4 バイナリ解析の結果を踏まえての対応

バイナリ解析では、ファームウェアやバイナリコードに含まれる情報を解析し、脆弱性や機密性の高い情報が含まれていないかを確認する。解析の結果として脆弱性が検出された場合には、脆弱性の修正や緩和策の検討が必要となる。脆弱性が攻撃された場合を想定した対策も有効である。例えば、バッファオーバーフロー脆弱性への対策機能を有効化することで、同脆弱性が攻撃された場合のセキュリティリスクを軽減することが可能である。また、第三者によるバイナリコードからの情報収集や制御フローの解析を防ぐための手段として、バイナリコードの難読化が有効である。これらの代表的な対応手法を以下に示す。

- **脆弱性を修正したソフトウェアコンポーネントに更新する**：検証対象機器が利用しているソフトウェアコンポーネントの脆弱性が検出された場合、機器メーカは対象となるソフトウェアコンポーネントの脆弱性を修正する。オープンソースソフトウェアやサードパーティ製のソフトウェアの場合には、脆弱性を修正済みの新しいバージョンに更新する。脆弱性を修正したソフトウェアが提供されていないケースや更新により機器の機能に影響があり脆弱性を製品のリリースまでに修正できない場合には、脆弱性に対する緩和策を検討すべきである。
- **バッファオーバーフロー対策機能を有効にする**：バイナリ解析によりバッファオーバーフロー脆弱性が判明した場合、バッファオーバーフロー脆弱性への対策として、スタック領域での命令実行

[24] モジュールの外部からのデータの読み取りや改ざんを困難にした機能

を防止する DEP[25]やプログラムのアドレス空間配置をランダム化する ASLR[26]等を有効化する。DEP を有効化したことによりバッファオーバーフロー脆弱性により例外処理が実行され、プロセスが停止することもあるため、適切に例外処理を行うようプログラムの修正も必要である。

- **バイナリコードを難読化する**：第三者によるバイナリ解析への対策として、バイナリコードを難読化する手法がある。本手法では、ソースコードをコンパイルする際に、バイナリコード内の制御フローや文字列データの複雑化や、デバッグ用のシンボル情報の削除を実施することでバイナリ解析を難しくする。また、デバッガの検知や動的なランダムなメモリ配置処理をバイナリコード内に挿入する等の難読化を行う手法、ツールもある。

5.5 ネットワークスキャンの結果を踏まえての対応

ネットワークスキャンでは、検証対象機器のネットワークサービスのサービスポートやネットワークサービスから収集可能な情報について検証する。機器メーカは、機器の仕様上不要なサービスや意図しないサービスが検証により判明した場合には、同サービスの無効化や通信制限を講じるべきである。また、ネットワークサービスから収集可能な情報については、必要な情報のみを出力するよう設定することが望ましい。これらの不備に対する代表的な対応手法を以下に示す。

- **不要なネットワークサービスを無効化する**：検証対象機器で動作する不要なネットワークサービスが判明した場合、デバッグ用に利用しているネットワークサービスや意図せず動作しているネットワークサービスについては、悪用される恐れがあるためサービスを停止・無効化する。なお、ネットワークサービスを無効化していたとしても脆弱性を突いて有効化され、悪用される可能性もあるため、検証対象機器が利用しないソフトウェアコンポーネントについては削除することが望ましい。
- **ファイアウォール機能を設定し、通信を制限する**：ネットワークサービスを無効化できない場合や必要なネットワークサービスの場合には、LAN からのアクセスのみに制限するなど、ネットワークインタフェースの用途に応じてファイアウォール機能で通信を制限する。
- **フィンガープリント情報を出力しない設定をする**：ネットワークサービスのレスポンスにおいて、動作しているソフトウェアの名称やバージョン情報のヒントが出力される場合、脆弱性検出へのヒントとなることがある。このため、ソフトウェアの名称やバージョン情報を出力しない設定をすることが望ましい。

5.6 既知脆弱性の診断の結果を踏まえての対応

脆弱性スキャンやパスワード検証によって、検証対象機器に含まれる脆弱性やパスワード情報を確認できる。脆弱性が検出された場合には、ソフトウェアの修正や緩和策の検討が必要となる。また、強度の低いパスワードが設定されている場合には、検証対象機器に求められるセキュリティレベルに合わせた内容のパスワードに修正するべきである。本内容に関する代表的な対応について以下に示す。

---

[25] Data Execution Prevention，メモリ上のプログラム領域以外のデータに関する領域でのプログラム実行を禁止する対策機能

[26] Address Space Layout Randomization，標準ライブラリなどのメモリアドレス空間の配置をランダム化する対策機能

- **脆弱性を修正したソフトウェアコンポーネントに更新する**：検証対象機器が利用しているソフトウェアコンポーネントの脆弱性が検出された場合、機器メーカは対象となるソフトウェアコンポーネントの脆弱性を修正する。オープンソースソフトウェアやサードパーティ製のソフトウェアの場合には、脆弱性を修正済みの新しいバージョンに更新する。脆弱性を修正したソフトウェアが提供されていないケースや更新により機器の機能に影響があり脆弱性を製品のリリースまでに修正できない場合には、脆弱性に対する緩和策を検討すべきである。
- **不要なソフトウェアコンポーネントを削除する**：検証対象機器の機能として利用しない不要なソフトウェアコンポーネントは、脆弱性が悪用された際のさらなる不正アクセスや踏み台攻撃として利用される恐れがある。例えば、tcpdump や libpcap 等のコンポーネントは、リモートコード実行脆弱性経由で通信データの盗聴に悪用されることがある。ソフトウェア BOM の構築結果を基に必要なコンポーネントと不要なコンポーネントを分類し、不要なコンポーネントは削除することが望ましい。
- **バッファオーバーフロー対策機能を有効化する**：バッファオーバーフロー脆弱性が判明した場合、バッファオーバーフロー脆弱性への対策として、スタック領域での命令実行を防止する DEP やプログラムのアドレス空間配置をランダム化する ASLR 等を有効化する。DEP を有効化したことによりバッファオーバーフロー脆弱性により例外処理が実行され、プロセスが停止することもあるため、適切に例外処理を行うようプログラムの修正も必要である。
- **強度が十分なパスワードの設定・認証メカニズムを有効にする**：パスワード検証により脆弱なパスワードの利用が検証された場合、パスワードの強度や認証メカニズムの見直しをする。英字や数字のみ、又は、8 文字以下で構成されるパスワード、よく利用されるパスワードは容易に解析されてしまうため、英数字、記号の組み合わせやパスワード長を長くする等のパスワード強度の高い設定をすることが望ましい。また、パスワードのハッシュアルゴリズムについては安全性が保証されたアルゴリズムやソルトの利用等、解析が難しい方式を利用することが望ましい。

5.7 ファジングの結果を踏まえての対応

ファジングの結果、脆弱性スキャンで発見できない新たな脆弱性が検出される場合がある。このような脆弱性は修正パッチが提供されていない場合もあり、脆弱性に関するソフトウェアの修正や緩和策を新たに検討する必要がある。これらの脆弱性や不備への代表的な対応手法として次の手法がある。

- **脆弱性を修正したソフトウェアコンポーネントに更新する**：検証対象機器が利用しているソフトウェアコンポーネントの脆弱性が検出された場合、機器メーカは対象となるソフトウェアコンポーネントの脆弱性を修正する。オープンソースソフトウェアやサードパーティ製のソフトウェアの場合には、脆弱性を修正済みの新しいバージョンに更新する。脆弱性を修正したソフトウェアが提供されていないケースや更新により機器の機能に影響があり脆弱性を製品のリリースまでに修正できない場合には、脆弱性に対する緩和策を検討すべきである。
- **バッファオーバーフロー対策機能を有効化する**：バッファオーバーフロー脆弱性が判明した場合、バッファオーバーフロー脆弱性への対策として、スタック領域での命令実行を防止する DEP やプロ

グラムのアドレス空間配置をランダム化する ASLR 等を有効化する。DEP を有効化したことによりバッファオーバーフロー脆弱性により例外処理が実行され、プロセスが停止することもあるため、適切に例外処理を行うようプログラムの修正も必要である。

- **ソフトウェアの入力チェックで不正な入力を防止する**：ファジングで検出される脆弱性は、検証対象機器が利用するフォーマットやプロトコルに違反するデータが入力されることに起因するという特徴がある。このようなデータに対し、ソフトウェアの入力チェックの段階で不正なデータを拒否することで脆弱性への攻撃を防止する対策となる。

5.8 ネットワークキャプチャの結果を踏まえての対応

ネットワークキャプチャでは、通信パケットを確認することで、暗号化通信の不備や機密性の高い情報、意図しない通信を検証できる。この結果として通信パケットの盗聴による情報漏えいやデータ改ざんによる不正操作等につながる恐れがある。これらの脆弱性や不備に対する代表的な対応手法を以下に示す。

- **通信の暗号化機能を正しく設定する**：デバッグ用の設定などでデジタル証明書を検証しない設定がされている場合や、期限切れのデジタル証明書を利用している場合があり、通信の暗号化が適切に行われないケースがある。このような不備に対して、デジタル証明書の検証設定や有効期限を確認することが望ましい。
- **機器メーカが意図しない通信の機能を無効化又はフィルタリングする**：OEM 製品や SDK を利用している機器の場合、機器メーカが意図しない通信が行われるケースがある。機器本来の機能として当該通信が不要な場合には、情報漏えいにつながる可能性もあるため、機能の無効化やファイアウォール機能によるフィルタリングを行うことが望ましい。
- **不要な Bluetooth プロファイルを無効化する**：Bluetooth には多様なプロファイルが定義されており、不要なプロファイルを受け付ける場合に情報漏えいや不正アクセスにつながる恐れがある。このため、検証対象機器の機能として不要なプロファイルについては無効化することが望ましい。

5.9 製品リリースにあたり機器メーカとして検討しておくべき事項

本編で記載したとおり、検証は機器の脆弱性を 100%網羅的に洗い出すものではない。検証によって問題が見つからなかった場合や、検証で発見された問題に対してリスク対応を行った場合でも、継続的なセキュリティ対策を講じることが必要となる。製品リリースにあたっては、対象となる機器に対して前節までの適切なセキュリティ対策を講じることに加え、機器メーカとして組織体制や運用においても検討しておくべき事項がある。本節では米国 NIST の NISTIR 8259[27]を参考とし、機器メーカとしての検討事項を示す。

NISTIR 8259 で示された機器メーカとしての検討事項を表 5-6 に示す。NISTIR 8259 では、製造段階に応じてフェーズを「IoT 機器等をリリースする前の活動」と、「IoT 機器等を市場にリリースした後の活動」に区分し、合計 6 つのアクティビティ（活動内容）を定義している。「IoT 機器等をリリースする

[27] NISTIR 8259 “Foundational Cybersecurity Activities for IoT Device Manufacturers” https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8259.pdf

前の活動については、本別冊の 3 章及び 4 章にて定義しているため、本節では「IoT 機器等を市場にリリースした後の活動」に該当する事項を示す。

表 5-6 NISTIR 8259 で示された機器メーカとしての検討事項

フェーズ	アクティビティ
IoT 機器等を市場にリリースする前の活動	アクティビティ 1： 予想される顧客とユーザを特定し、想定されるユースケースを定義する
	アクティビティ 2： 顧客のサイバーセキュリティのニーズと目標を調査する
	アクティビティ 3： 顧客のサイバーセキュリティ目標に対処する方法を決定する
	アクティビティ 4： 顧客のサイバーセキュリティ目標に対する適切なサポートを計画する
IoT 機器等を市場にリリースした後の活動	アクティビティ 5： 顧客とのコミュニケーション方法を定義する
	アクティビティ 6： 顧客に何を伝えるか、どのように伝えるかを決定する • IoT 機器等のサポート期間と製品寿命 • IoT 機器等の構成と機能 （ソフトウェア、ファームウェア、ハードウェア、サービス、機能、データタイプなどの情報） • IoT 機器等のソフトウェア、ファームウェアの更新情報 • IoT 機器等をセキュアに廃棄するためのオプション情報 • IoT 機器等が提供するサイバーセキュリティ機能

5.9.1 利用ユーザへの情報提供について

機器メーカは、ユーザに対して、安心、安全に製品を利用する上で必要な情報を提供する必要がある。利用者に情報を提供する際には、アクセスが容易であり、かつ情報が視認しやすいよう配慮すべきである。以下に利用者への情報提供に関するポイントを示す。

- **対象機器のサポート期限や方針を明確にする：**対象機器のサポートは、ソフトウェアのアップデートを含め、いつまで提供されるのかを明確化する。また、サポート終了時には、どのようなタイミングで利用者に通知され、対象機器はどのように取り扱われるのかを明確化しておく。特に、サポート終了後にセキュリティに対して影響が生じる可能性がある場合には、注意喚起も含めて、利用者への周知を徹底する必要がある。またサポート終了後も、継続的に利用可能な機能がある場合は、機器メーカ側の免責事項も含めて、利用者へ情報提供を行う。
- **対象機器のセキュリティに影響する機能を明示する：**対象機器の機能において、セキュリティに影

響がある設置や設定や構成については利用者に提示し、脆弱な設定や環境構成で利用されないよう注意喚起を促す。特に、設定や環境構成の変更によってセキュリティ上重大なリスクが懸念される場合には、具体的なリスクの内容を示し、相談先のサポート窓口の明記や免責事項の提示を行う。

- **対象機器が収集する情報及び情報の取り扱いに関するポリシーを提示する：**対象機器が収集する情報については、利用者にとって分かりやすい表現で提示する必要がある。特に機密性の高い情報や、個人情報、プライバシー情報を収集する場合には利用者へ明示する。また、収集された情報は、誰がどのような目的で利用するのを明確化することに加え、機器メーカとして情報の取り扱いに関するポリシーを文書化し、利用者へ提示する。
- **ソフトウェアのアップデート方法を利用者へ伝える：**対象機器のソフトウェアのアップデート方法については、下記の例を参考に利用者へ情報を提示する。アップデート時に利用者へ伝える情報の例として以下の項目が挙げられる。
  - 利用可能なアップデートソフトウェアが存在するかどうか
  - アップデートの内容：機能追加や変更なのか、脆弱性や不具合に対する修正か
  - 不具合や脆弱性の情報：問題が発生するソフトウェアのバージョンや利用者への影響（アップデートを行わない場合のリスクなど）
  - アップデートの方法/手順：自動更新なのか手動更新が必要か
    （手動更新の場合はアップデートソフトウェアの入手先（Web リンクや URL）や更新手順を提示する。バックアップが必要な機器については、バックアップ方法も含めて提示を行う。）
  - アップデートを実施すべき主体：利用者が行うのか、製品提供側（保守担当者など）で行う必要があるか
- **対象機器の廃棄方法を明確にする：**対象機器を廃棄あるいは譲渡する場合の対処方法を利用者へ提示する。特に利用者によって機密性の高い情報が蓄積される機器については、情報の初期化方法や、情報が残存したまま廃棄された場合のリスク、機器メーカとしての免責事項などを明確化する。

5.9.2 セキュリティサポートの体制について

機器メーカは利用者への情報提供に加え、セキュリティサポート体制を構築し、対象製品販売後も継続的に安心、安全に機器が利用されるよう運用を行う必要がある。

- **対象機器のセキュリティサポートの体制を整備する：**対象機器において発生した問題に対処するため、機器メーカとしてセキュリティサポートの体制を構築する必要がある。セキュリティサポートとしては、利用者に対する相談窓口の設置や、脆弱性情報の収集、不具合や脆弱性に対するソフトウェアアップデートの提供等を行う体制が必要となり、機器の開発部門等と連携した体制を整備することが望まれる。また、対象機器が複数企業のサプライチェーンによって提供されている場合には、責任範囲の明確化や、ユーザのセキュリティサポートを提供する主体の明確化など、企業間による事前合意や連携した取り組みが必要となる。

- **脆弱性及びサイバーセキュリティへの影響が懸念される問題の報告先を明示する：**対象機器において脆弱性やサイバーセキュリティへの影響が懸念される問題が見つかった場合に、報告先となる窓口の情報を公開する。こうした情報は利用者だけでなく、ホワイトハッカーなど、善意の第三者からも有益な情報が得られる可能性があるため、脆弱性届出制度[28]などを活用し、外部機関と連携した対応を行うことが望ましい。
- **インシデントや脆弱性発見時の組織体制やポリシーを明確にする：**対象機器において、インシデントや脆弱性が発見された場合に備え、インシデントレスポンス体制を行うための組織体制やポリシーを整備する。企業に対するサイバー攻撃に対応する機能として CSIRT（Computer Security Incident Response Team）を組織することに加え、製品に対応する機能である PSIRT（Product Security Incident Response Team）を組織し、インシデントの原因究明や証拠保全、再発防止に向けた対応を行うことが望ましい。また効果的にインシデントの再発防止を図るためには、JPCERT/CC の連携に加え、サプライチェーンに関するインシデントや脆弱性への対応については、ステークホルダー企業間での連携や、関連業界おいてインシデントの共有を図る各 CERT 機関との外部連携を行うことが求められる。

5.9.3 次期製品開発へのフィードバック

検証サービス事業者による検証結果報告を受けて、次期製品開発のセキュリティ対策に対してフィードバックすることが望まれる。特に重要なフィードバックが事項を以下に示す。

- **次期製品開発時のセキュリティ検証において、過去の検証結果を反映する**：次期製品開発時のセキュリティ検証においては、過去の検証結果を踏まえて検証を依頼・実施する。過去の製品に対する検証において脆弱性が検出された場合には、当該脆弱性が内在した箇所や当該脆弱性に関わる検証手法について、優先度を高めることが望まれる。
- **ソフトウェア構成表（ソフトウェア BOM）を管理、更新する**：IoT 機器等に使用されるアプリケーションは、様々なソフトウェア群（コンポーネント）によって構成される。各コンポーネントは市販のソフトウェアやドライバ、フレームワークだけでなく、自社で開発されたコードやオープンソースで開発されたものも含めて、複雑なサプライチェーンによって提供されている。その中には、検証によって脆弱性が検出されたソフトウェアが含まれるケースもあり、最新バージョンへの更新が必要となる。機器メーカは、対象製品のコンポーネントに含まれるソフトウェアの名称やバージョンなどをソフトウェア構成表（ソフトウェア BOM）として管理し、更新を適用した場合には構成表の更新を行う必要がある。対象機器での対応が困難であり、次期開発において対応する場合にも、ソフトウェア構成表を基に設計段階から検討を行うことで、より効率的な設計開発を行うことができる。
- **要求仕様や設計のフェーズでのセキュリティ対応を行う**：検証結果によっては、セキュリティ課題の原因が要求仕様や設計に起因し、機器自体での対応が困難なケースもある。こうした場合には、第 5.1 節で示した対応記録にて課題管理を行い、次機種の企画・要件定義フェーズから、

[28] IPA 脆弱性関連情報の届出受付 https://www.ipa.go.jp/security/vuln/report/

設計フェーズにて対応を行う。また原因を作り込んでしまった原因を調査し、ドキュメントレビューやコードレビューの方針是正など、設計開発プロセス全体を意識した改善活動を行うことが求められる。

6 付録

6.1 国内外のセキュリティガイドラインと本別冊との対応

6.1.1 国内外のセキュリティ対策動向

図 6-1 に示すとおり、IoT 機器等に関連するセキュリティ規格が、我が国をはじめ米国、欧州からも提案がなされている。我が国では経済産業省が IoT 機器等及びサプライチェーンを対象とするセキュリティ対策の指針として、2019 年 4 月に「サイバー・フィジカル・セキュリティ対策フレーム（CPSF）」を策定した。このフレームワークでは、産業・社会の変化に伴うサイバー攻撃の脅威の増大に対し、リスク源を適切に捉え、検討すべきセキュリティ対策を漏れなく提示するための新たなモデルを三層構造と 6 つの構成要素として提示し、それぞれにおいて守るべきもの、直面するリスク源、対応方針等を整理した。

米国では、2018 年 9 月にカリフォルニア州法として、「コネクテッド・デバイスのセキュリティに関する法律」（SB327）が制定され、2020 年 1 月には施行されている。また IoT 機器等に対する政府調達基準についても具体的な検討が進められており、2019 年 3 月には法案「Internet of Things Cybersecurity Improvement Act of 2019」が提出され、さらにより具体的なガイドラインとして「NISTIR 8259」のファイナル版が 2020 年 5 月に公開された。2020 年 11 月には、上記法案が上院を通過した。

欧州では、2018 年 10 月に英国政府による「消費者向け IoT 製品のセキュリティに関する行動規範」[29]が公開され、IoT 機器等のセキュリティに関する基本対策項目（13 項目）が提示された。また欧州全体としてもこの行動規範を基に「ETSI EN 303 645」が 2020 年 6 月に最終版として公開された。現状は推奨事項であるが、将来的には法制化をも視野に入れた規格であり、引き続き今後の動向の注視が必要となる。

29 https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/775860/054718_DCMS_IoT_Code_of_Practice_JAPANESE.pdf

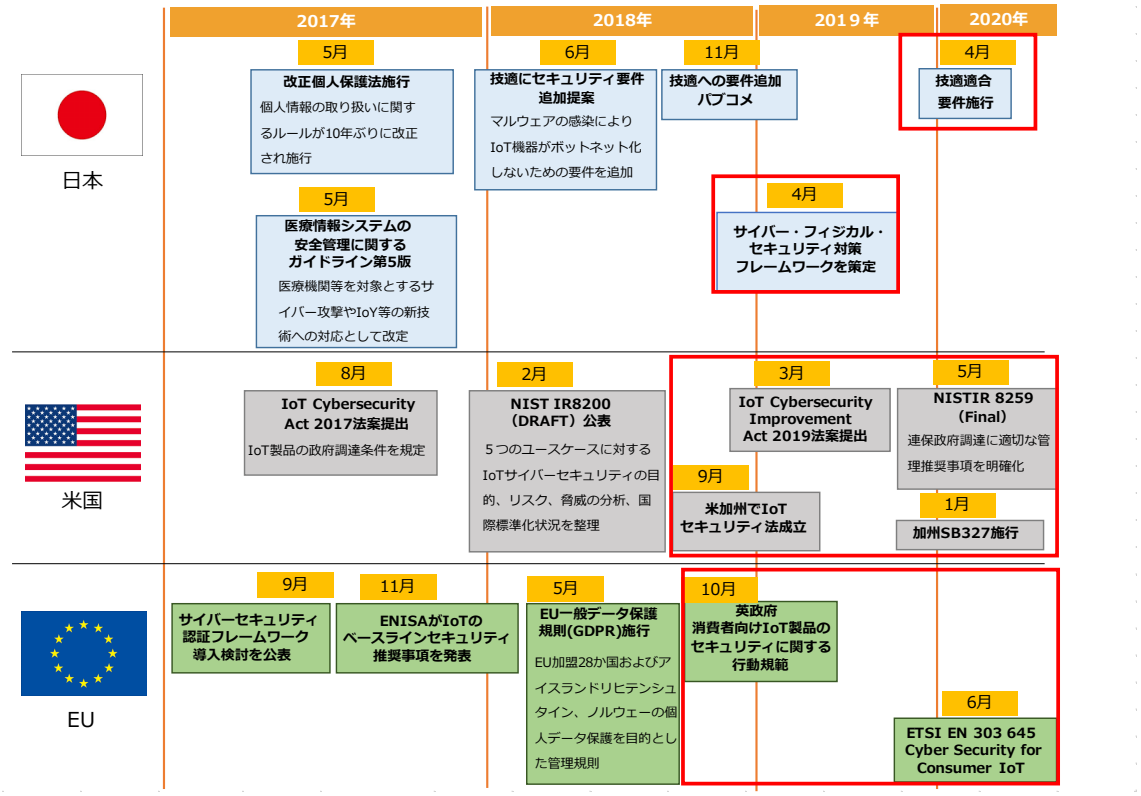

図 6-1　IoT 機器等に関連するセキュリティ規格の国際動向 [30]

6.1.2 本別冊と他の IoT 向けセキュリティガイドラインとの対応関係

本別冊におけるセキュリティ対応事項と国内外の IoT 向けセキュリティガイドラインとの対応を表 6-1 に示す。

表 6-1　本別冊と他の IoT 向けセキュリティガイドラインとの対応関係

本別冊におけるセキュリティ対応事項	日 CPSF	米 NISTIR 8259	EU ETSI 303 645	CCDS GR01- 2021[31]
5.3 ハードウェアハッキング・ファームウェア解析の結果を踏まえての対応				
不要なデバッグインタフェースを無効化する	–	–	○	–
ファイルシステムを暗号化する	○	○	○	–
5.4 バイナリ解析の結果を踏まえての対応				

[30] 一般社団法人重要生活機器連携協議会（CCDS）公開資料より加筆修正
https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/dainiso/pdf/002_04_00.pdf

[31] CCDS「IoT 分野共通セキュリティ要件ガイドライン 2021 年版（CCDS-GR01-2021）」
https://www.ccds.or.jp/public/document/other/CCDS_SecGuide-IoTCommonReq_2021_v1.0_jpn.pdf

本別冊におけるセキュリティ対応事項	日 CPSF	米 NISTIR 8259	EU ETSI 303 645	CCDS GR01- 2021[31]
脆弱性を修正したソフトウェアコンポーネントに更新する	○	○	○	–
バッファオーバーフロー対策機能を有効にする	–	–	–	–
バイナリコードを難読化する	–	–	–	–
5.5 ネットワークスキャンの結果を踏まえての対応				
不要なネットワークサービスを無効化する	○	○	○	○
ファイアウォール機能を設定し、通信を制限する	○	○	○	○
フィンガープリント情報を出力しない設定をする	–	–	–	–
5.6 既知脆弱性の結果を踏まえての対応				
不要なソフトウェアコンポーネントを削除する	○	–	–	–
強度が十分なパスワードの設定・認証メカニズムを有効にする	○	○	○	○
5.7 ファジングの結果を踏まえての対応				
ソフトウェアの入力チェックで不正な入力を防止する	–	–	○	○
5.8 ネットワークキャプチャの結果を踏まえての対応				
通信の暗号化機能を正しく設定する	○	○	○	–
機器メーカが意図しない通信の機能を無効化又はフィルタリングする	○	–	○	–
不要な Bluetooth プロファイルを無効化する	–	–	–	○
5.9 製品リリースにあたりメーカとして検討しておくべき事項				
対象機器のサポート期限や方針を明確にする	–	○	–	–
対象機器のセキュリティに影響する機能を明示する	–	○	○	–
対象機器が収集する情報及び、情報の取り扱いに関するポリシーを提示する	–	○	○	–
ソフトウェアのアップデート方法を利用者へ伝える	–	○	○	○
対象機器の廃棄方法を明確にする	–	○	–	–
対象機器のサポートの体制を整備する	–	○	–	○
脆弱性及びサイバーセキュリティへの影響が懸念される問題の報告先を明示する	○	○	–	○

本別冊におけるセキュリティ対応事項	日 CPSF	米 NISTIR 8259	EU ETSI 303 645	CCDS GR01- 2021[31]
インシデントや脆弱性発見時の組織体制やポリシーを明確にする	○	–	–	○
ソフトウェア構成表（ソフトウェア BOM）を管理、更新する	○	–	–	–
次機種開発へのフィードバックを行う	○	–	–	–

6.2 検証依頼時に用意することが望まれる情報の逆引き表

第 4.8 節で紹介した検証依頼時に用意することが望まれる情報の参考情報として、逆引きの情報を付録の第 6.2 節に示す。

表 6-2 検証依頼時に用意することが望まれる情報の逆引き

分類	情報	関連手法
ハードウェア	チップのデータシート（メーカ、型番、ピン配置）	• ハードウェアの情報収集
	デバッグインタフェース情報（デバッグインタフェースの種別、ピン配置）	• デバッグインタフェース（JTAG，UART）の特定とアクセス可否の確認
	メモリアドレス情報（RAM、フラッシュメモリ）	• デバッグインタフェースからのファームウェア抽出
	ネットワークインタフェース	• ネットワーク経由の脆弱性スキャン • ネットワークポートスキャン
ソフトウェア	ファームウェアイメージファイル	• ファームウェア解析 • ソフトウェア BOM の構築・脆弱性スキャン • ハードコーディングパスワードの調査
	バイナリコード	• バイナリ解析とバッファオーバーフロー脆弱性の特定 • シンボリック実行を利用したバイナリ解析 • ハードコーディングパスワードの調査 • ファイルフォーマット・ネットワークプロトコルベースのファジング • コード網羅率を指標とするファジング
	パスワードファイル	• パスワードファイルの検証
	バイナリコードのデバッグ・シンボル情報	• バイナリ解析とバッファオーバーフロー脆弱

分類	情報	関連手法
		性の特定 • シンボリック実行を利用したバイナリ解析 • ファイルフォーマット・ネットワークプロトコルベースのファジング • コード網羅率を指標とするファジング
	検証対象機器のモバイルアプリケーションデータ	• APK ファイルのデコンパイル
	バイナリコードのソースコード	• バイナリ解析とバッファオーバーフロー箇所の特定 • シンボリック実行を利用したバイナリ解析 • ファイルフォーマット・ネットワークプロトコルベースのファジング • コード網羅率を指標とするファジング
	ネットワークサービス情報（サーバソフトウェア、バージョン、プロトコル、ポート番号）	• ネットワーク経由の脆弱性スキャン • ネットワークポートスキャン • アプリケーション通信のパケットキャプチャ • ファイルフォーマット・ネットワークプロトコルベースのファジング • コード網羅率を指標とするファジング
	認証情報（ユーザ ID、パスワード）	• ハードコーディングパスワードの調査 • パスワードファイルの検証 • ネットワーク経由の脆弱性スキャン
	Bluetooth の通信情報（Bluetooth バージョン、認証方式、プロファイル、アプリケーションプロトコル）	• Bluetooth パケットキャプチャ
その他ドキュメント類	ユーザマニュアル	※ 検証手法全般で利用

6.3 用語集

- **ASLR（Address Space Layout Randomization）**
  標準ライブラリなどのメモリアドレス空間の配置をランダム化するバッファオーバーフロー対策機能。

- **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、0～10.0 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。

- **DEP（Data Execution Protection）**
  メモリ上のプログラム領域以外のデータに関する領域でのプログラム実行を禁止するバッファオーバーフロー対策機能。

- **IoT（Internet of Things）**
  既存もしくは開発中の相互運用可能な情報通信技術により、物理的もしく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 機器等**
  IoT を構成する、ネットワークに接続される機器。

- **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。

- **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。

- **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。

- **STRIDE**
  Spoofing（なりすまし）、Tampering（改ざん）、Repudiation（否認）、Information Disclosure（情報漏えい）、Denial of Service（サービス拒否）、Elevation of Privilege（権限昇格）の六つの脅威の性質の頭文字から構成され、これら六点の性質から脅威を洗い出していく手法。

- **JVN (Japan Vulnerability Notes)**
  日本で使用されているソフトウェアなどの脆弱性関連情報とその対策情報を提供し、情報セキュリティ対策に役立てることを目的とした脆弱性対策情報ポータルサイト。JPCERT/CC と情報処理推進機構（IPA）が共同で管理。

- **NVD (National Vulnerability Database)**

NIST が管理する脆弱性情報データベースで、他のデータベース機関へのリンクに留まらず、関連する外部サイト情報も提供する。

- **TLPT（Threat-Led Penetration Test）**
  実在の攻撃者の戦術、テクニック、手順等を模倣し、組織のサイバーレジリエンスを侵害しようとすることを目的としたペネトレーションテスト。攻撃側（Red Team）の脅威情報に基づく現実的な攻撃に対して、防御側（Blue Team）は組織として防御、検知、対応等を行い、組織全体のレジリエンス能力を評価する。

- **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。

- **エントロピー解析（Entropy Analysis）**
  データの情報量（エントロピー）を解析して定量化する手法。圧縮や暗号化されたデータの場合は高い値を示す。

- **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]

- **脅威情報（Threat Intelligence）**
  脅威からの保護、攻撃者の活動検知、脅威への対応等に役立つ可能性のある情報。[NIST SP 800-150]

- **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もしくは使用の試み。[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。マルウェア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]

- **シルク（Silk）**
  部品やピンの配置を分かりやすくするためにプリント基板に印字したテキストや記号。

- **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。

- **セキュリティ・バイ・デザイン（Security by Design）**
  情報セキュリティを企画・設計段階から確保するための方策。[内閣サイバーセキュリティセンター]
  IoT 機器等においても、製品の企画・設計のフェーズからセキュリティ対策を組み込み、サイバーセキュリティ対策を確保しておく概念として、適用することができる。

- **耐タンパ性（Tamper Resistance）**
  モジュールの外部からのデータの読み取りや改ざんを困難にした機能。

- **テイント解析（Taint Analysis）**
  バイナリコードにデータを入力した際に当該データに関連するデータフローを解析する手法。

- **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]

- **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マルウェア、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]

- **リプレイ攻撃（Replay Attack）**
  ネットワークの通信データを傍受し、同じ内容のデータを再送信する攻撃手法。リプレイ攻撃に対して脆弱なシステムの場合、暗号や認証に必要な情報を第三者が分からない状態でなりすましできる可能性がある。

- **レジリエンス（Resilience）**
  システムが以下の状態を維持できること：①悪条件下にあっても、あるいは負荷がかかった状態であっても、（顕著に低下した状態又は無力化したような状態に陥ったとしても）稼働して、基礎的な運用能力を維持すること。②ミッションニーズと平仄が合う時間内に、有効的に運用されている状態に復旧すること。[NIST SP 800-53 Rev.4]

6.4 参考文書

- **サイバー・フィジカル・セキュリティ対策フレームワーク（経済産業省）**
  https://www.meti.go.jp/press/2019/04/20190418002/20190418002-2.pdf

- **IoT 開発におけるセキュリティ設計の手引き （IPA）**
  https://www.ipa.go.jp/files/000052459.pdf

- **IoT セキュリティ評価検証ガイドライン 第 1 版（CCDS）**
  https://www.ccds.or.jp/public_document/index.html#Verification_guidelines1.0

- **組込みソフトウェア開発における品質向上の勧め[バグ管理手法編]（IPA）**
  https://www.ipa.go.jp/sec/softwareengineering/std/emb.html

- **脆弱性対処に向けた製品開発者向けガイド（IPA）**
  https://www.ipa.go.jp/security/vuln/report/notice/guideforvendor.html

- **IoT セキュリティ・セーフティ・フレームワーク（経済産業省）**
  https://www.meti.go.jp/press/2020/11/20201105003/20201105003.html
- **製品分野別セキュリティガイドライン_スマートホーム編 1.0 版（CCDS）**
  https://www.ccds.or.jp/public_document/index.html#guidelines-smarthome_1.0
- **NISTIR 8259 “Foundational Cybersecurity Activities for IoT Device Manufactures”（NIST）**
  https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8259.pdf
- **NISTIR 8259A “IoT Device Cybersecurity Capability Core Baseline”（NIST）**
  https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8259A.pdf
- **ETSI EN 303 645 ”Cyber Security for Consumer Internet of Things: Baseline Requirements”（ETSI）**
  https://www.etsi.org/deliver/etsi_en/303600_303699/303645/02.01.01_60/en_303645v020101p.pdf