JISEC

2013 年3 月4 日発行

開発者のためのセキュア解説書（脆弱性評価編）

IT製品のセキュリティ機能が正しく動くことと同様に、そのIT製品に対する攻撃に悪用される欠陥（脆弱性）をなくすことが開発者にとって大きな課題です。

ITセキュリティ評価の国際的な規格であるコモンクライテリアでは、それらの脆弱性評価も規定されています。本書では、脆弱性の情報探索や脆弱性の確認のためのテストを実施する際の注意事項を解説します。

独立行政法人
情報処理推進機構

技術本部セキュリティセンター
情報セキュリティ認証室
www.ipa.go.jp/security/jisec/

ー 目次 ー

1 はじめに

IT 製品において、攻撃に悪用される可能性のある欠陥（脆弱性）をなくすことは、開発者にとって大きな課題となっています。

IT セキュリティ評価の国際的な規格であるコモンクライテリア（Common Criteria、以下「CC」といいます。）には、評価対象の IT 製品のセキュリティ機能を検査し、悪用可能な脆弱性が存在しないことを保証するための評価方法が規定されています。本書は、CC の脆弱性評価方法を紹介し、脆弱性を検出するための探索や、攻撃に悪用される可能性を確認するための侵入テストを実施する際の注意点を解説します。

CC の脆弱性評価の考え方は、CC 評価を行う評価者だけでなく、開発者内部での IT 製品のセキュリティに関するレビューや品質テストにも有効です。本書が、CC の理解とともに、IT 製品のセキュリティを向上させるための各種取り組みにおいて、参考になれば幸いです。

1.1 本書の対象読者

本書の対象読者は、IT 製品のセキュリティ機能の設計実装やテストに携わる開発者や、CC に基づいて脆弱性評価を行う評価者を想定しています。

本書は、CC 規格を解説しているため、CC に基づいて評価を行う「評価者」を主体に記述されています。しかし、脆弱性評価の内容は、開発の過程で考慮されるべき内容です。開発者の方は、本書に記述されている「評価者」を開発者自身と捉えることにより、本書の内容を開発者自身の IT 製品開発に適用することができます。

1.2 本書の構成

本書は以下の 6 章で構成されています。

■ 1 章　はじめに
　本書の目的や対象読者について説明しています。

■ 2 章　脆弱性評価の概要
　CC の脆弱性評価の全体的な概要を説明しています。

■ 3 章　攻撃能力
　CC の脆弱性評価の合否の判定基準である「攻撃能力」を、具体的な攻撃シナリオとその攻撃能力の算出例を示しながら説明しています。

■ 4章　脆弱性探索

CC で要求されている脆弱性探索の概要と、実際に脆弱性を探索する際の参考情報や注意点を説明しています。

■ 5章　侵入テスト

想定した脆弱性が実際に存在するか否かを決定するための侵入テストについて、参考情報や注意点を説明しています。

■ 6章　おわりに

本書で解説している内容について、全体をとおしての注意点を説明しています。

1.3 コモンクライテリア規格文書

本書の評価基準及び評価方法は、以下の表 1-1 及び表 1-2 の規格文書に基づいています。評価基準は「CC」、評価方法は「CEM」という略称で呼ばれています。

表 1-1　CC/CEM の規格文書（日本語翻訳版）

CC / CEM バージョン 3.1 リリース 4（CC / CEM v3.1 Release4）
評価基準 情報技術セキュリティ評価のためのコモンクライテリア （CC バージョン 3.1 リリース 4）
パート 1: 概説と一般モデル　バージョン 3.1 改訂第 4 版［翻訳第 1.0 版］
パート 2: セキュリティ機能コンポーネント バージョン 3.1 改訂第 4 版［翻訳第 1.0 版］
パート 3: セキュリティ保証コンポーネント バージョン 3.1 改訂第 4 版［翻訳第 1.0 版］
評価方法 情報技術セキュリティ評価のための共通方法 （CEM バージョン 3.1 リリース 4）
評価方法　バージョン 3.1 改訂第 4 版［翻訳第 1.0 版］

表 1-2　CC/CEM の規格文書（原文）

CC / CEM v3.1 Release4
評価基準 Common Criteria for Information Technology Security Evaluation （CC v3.1 Release4）
Part 1:　Introduction and general model　Version 3.1　Revision 4
Part 2:　Security functional components　Version 3.1　Revision 4
Part 3:　Security assurance components　Version 3.1　Revision 4
評価方法 Common Methodology for Information Technology Security Evaluation （CEM v3.1 Release4）
Evaluation methodology　Version 3.1　Revision 4

本書は、CC/CEM の規格文書のうち、CEM[1]の以下の記載内容に基づいています。

- CEM,「15　AVA クラス: 脆弱性評定」
- CEM,「附属書 B　脆弱性評定(AVA)」

1.4 用語集

本書で使用する CC/CEM に関係する用語を表 1-3 に示します。

表 1-3 CC/CEM 用語集

用語	説明
CC (Common Criteria：コモンクライテリア)	情報セキュリティの観点から、IT 製品が適切に設計され、その設計が正しく実装されていることを評価するための国際標準規格「ISO/IEC 15408」のことです。
CEM (Common Evaluation Methodology：共通評価方法)	CC に基づいたセキュリティ評価を均質に行うために定められた評価の方法のことです。CC 規格を満たすための、評価すべき項目や評価の観点が定められています。
EAL (Evaluation Assurance Level：評価保証レベル)	CC に基づいたセキュリティ評価の保証の度合いを表します。EAL1 から EAL7 の 7 段階が定義されています。EAL が高くなるほど、製品の広範囲の設計情報等が厳格に評価されます。
セキュリティターゲット	CC に基づいたセキュリティ評価のために、評価対象の IT 製品について記述した文書のことです。IT 製品の評価範囲、前提条件、評価対象のセキュリティ機能、評価保証レベルなどが記述されています。セキュリティターゲットは、IT 製品の調達者の要求に基づいて、IT 製品の開発者が作成します。

2　脆弱性評価の概要

本章では、CC の脆弱性評価について、全体的な評価の概要を説明します。

2.1 CC の脆弱性評価の特徴

CC に基づく IT 製品のセキュリティ評価では、評価者は、IT 製品及びその設計書や利用者向けガイダンスなどの証拠資料を検査して、セキュリティ機能が正確に実装されているかどうかと共に、脆弱性が存在しないかどうかを評価します。

脆弱性評価については、CC 評価以外にも、商用のセキュリティ検査サービスが数多く存在します。しかし、それらの検査の手法や品質は、サービス提供者によって異なります。一方、CC では、認定された評価機関の評価者が検査すれば同じ結果が導き出せるように、評価方法が決められています。また、その評価方法では、製品のセキュリティ機能に対して、評価者が製品の調達者が必要とする以上の過度な要求をしないように、配慮されています。

そのため、CC の脆弱性評価は、一般的なセキュリティ検査サービスと異なる部分があります。主な違いは以下のとおりです。

■開発者設計情報の考慮

一般的なセキュリティ検査サービスでは、主として、既に知られている脆弱性情報に基づいてブラックボックステストが行われています。一方、CC では、評価者は、既に知られている脆弱性情報に加えて、開発者の提示する設計書等も検査した上で、製品に存在する可能性のある脆弱性を洗い出し、テストで確認します。

開発者が提示しなければならない証拠資料は、製品の調達者の要求する評価保証レベル(セキュリティターゲットで指定された EAL)毎に決められています。例えば、製品の設計書の場合、EAL1 では外部インタフェース仕様のみ、EAL2 以上では製品内部の設計が追加され、EAL4 以上ではさらにソースコードが追加されるなど、EAL が高くなるほど広範囲の証拠資料が詳細に評価されることになります。

■攻撃に必要な能力（攻撃の困難さ）の考慮

一般に、製品のセキュリティ機能がどの程度の攻撃に耐えられれば良いかは、製品の調達者の意図する用途などによって異なります。CC ではそれを考慮して、評価で検出された脆弱性に対して、実際の攻撃に悪用する困難さと製品の調達者の要求を考慮して合否が判定されます。

具体的には、CC では、評価対象製品が耐えなければならない攻撃者の能力が、調

達者の要求する評価保証レベル（セキュリティターゲットで指定されたEAL）毎に決められています。例えば、評価対象製品に脆弱性が存在したとしても、攻撃者がそれを悪用して攻撃を成功させるために必要な攻撃者の能力が、EAL毎に決められた基準値を上回る（つまり、実際にその脆弱性を悪用して攻撃を成功させることは困難である）場合には、評価対象製品は攻撃に対して必要な耐性を備えていることになり、CC評価は合格となります。

■前提条件の考慮

CCでは、製品の調達者の運用環境に対する前提条件（セキュリティターゲットで指定された前提条件）が考慮されます。例えば、評価対象製品に技術的には脆弱性とみなされる問題点が存在したとしても、前提条件に従った運用対策で、その問題点を悪用した攻撃を防止できる場合には、CC評価は合格となります。

■脆弱性の限定

CCでは、脆弱性は、製品の調達者の要求するセキュリティ機能（セキュリティターゲットで評価の対象として指定されたセキュリティ機能）の侵害を意味します。例えば、評価対象製品にサービス妨害を引き起こす問題が存在したとしても、セキュリティターゲットで指定されたセキュリティ機能の侵害にならない場合には、CC評価上は脆弱性とはみなされません。

2.2 CC の脆弱性評価の流れ

CC に基づく脆弱性評価の流れを図 2-1 に示します。

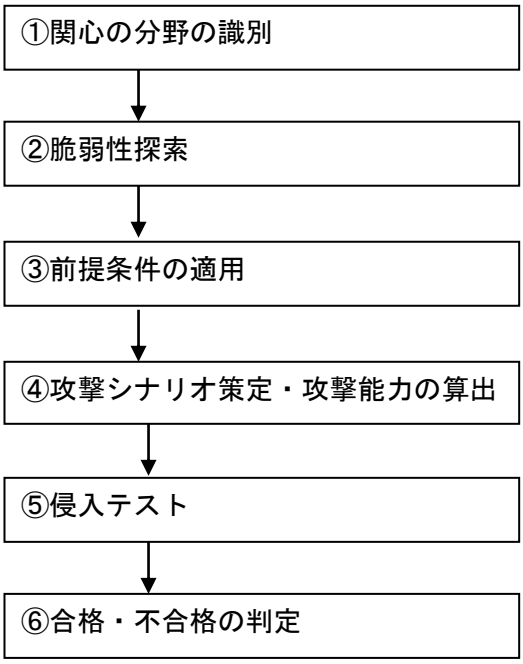

図 2-1　脆弱性評価の流れ

図 2-1 の項目の概要は以下のとおりです。

①関心の分野の識別

「関心の分野」は、評価対象製品の設計や実装の中で、評価者が脆弱性の詳細な調査の必要性を判断した部分のことです。CC 評価では、評価者は、脆弱性分析に先立って、セキュリティターゲット、設計書、利用者向けガイダンスなどの証拠資料を検査します。その検査において、評価者はセキュリティ上の問題が発生する可能性のある処理を識別しておきます。その処理部分は「関心の分野」として、以降の脆弱性探索の入力となります。

②脆弱性探索

評価者は、評価対象製品の「関心の分野」や採用技術等の情報を基に、評価対象製品に存在する可能性のある脆弱性を探索してリストアップします。脆弱性の探索は、インターネットの検索エンジン等を用いた一般に知られている脆弱性（これを「公知の脆弱性」といいます。）の探索と、評価対象製品の証拠資料の分析を組み合わせて行われます。

なお、評価者は、脆弱性探索のために証拠資料を分析している際にも、セキュリ

ティ上の問題が発生する可能性のある別の「関心の分野」を新たに識別する場合があります。その場合、評価者は、新たに識別した「関心の分野」についても、公知の脆弱性探索と証拠資料の分析を組み合わせて、脆弱性を探索します。

③前提条件の適用

脆弱性探索でリストアップされた脆弱性の可能性について、評価者は、前提条件が守られた運用環境の評価対象製品にあてはまるかどうかを分析します。技術的には製品に存在する可能性のある脆弱性であっても、前提条件が守られた運用環境では発生することのない脆弱性は、この段階で分析の対象から除外されます。残りの脆弱性は、以降の分析の対象となります。

④攻撃シナリオ策定・攻撃能力算出

評価対象製品に存在する可能性のある脆弱性について、評価者は、脆弱性を悪用した攻撃のシナリオを立案し、その攻撃シナリオの実行に必要な攻撃者の能力（これを「攻撃能力」といいます。）を算出します。攻撃シナリオは、以降の侵入テストで確認されます。なお、脆弱性の悪用に必要な攻撃能力が、EAL 毎に決められた基準値を明らかに上回る（つまり、攻撃を成功させることが明らかに困難である）場合には、侵入テストを実施する必要はありません。

⑤侵入テスト

評価者は、攻撃シナリオに基づいてテスト項目を立案し、侵入テストを実施して、想定した脆弱性が評価対象製品に実際に存在するかどうかを検査します。

⑥合格・不合格の判定

前提条件の守られた運用環境の評価対象製品に対して、適用可能な脆弱性が存在しない場合や、侵入テストが成功しなかった場合には、CC 評価は合格となります。

評価対象製品に対して侵入テストが成功した場合、想定した脆弱性が実際に存在することになります。この場合の CC 評価の合格・不合格は、その脆弱性を悪用するために必要な攻撃能力の値と、EAL 毎に決められた基準値を、比較して決定されます。評価者は、侵入テストを立案し攻撃が成功するまでの労力を考慮して、攻撃能力の値を再度算出し、合否を判定します。

3 攻撃能力

本章では、CC の脆弱性評価の合否判定に使用される攻撃能力について説明します。

3.1 攻撃能力の定義

攻撃能力は、脆弱性を悪用するための攻撃シナリオについて、その攻撃シナリオを実行するために必要な攻撃者の能力を数値化したものです。攻撃能力は、表 3-1 の 5 つの要因毎に算出した数値の合計値で表されます。なお、表 3-1 の「攻撃」には、攻撃者が脆弱性を発見し、その脆弱性を悪用するための攻撃方法を考案し、実際に攻撃を成功させるまでの、攻撃者のすべての労力が含まれています。

表 3-1 攻撃能力の要因

要因	内容
所要時間	攻撃に必要な時間を意味します。「1 日未満」(値 0)、「1 日～1 週間」(値:1)、「1 週間～2 週間」(値:2)、「2 週間～1 ヶ月」(値:4)など、所要時間に応じて重み付けされています。
専門知識	攻撃に必要な一般的な技術的知識を意味します。「しろうと」(値:0)、「熟練者」(値:3)、「エキスパート」(値:6)など、知識のレベルに応じて重み付けされています。
評価対象の知識	攻撃に必要な対象製品の設計や運用の知識を意味します。「公開情報」(値:0)、「限定的情報」(値:3)、「機密情報」(値:7)など、製品情報の入手の困難さに応じて重み付けされています。
機会	攻撃に必要な対象製品へのアクセス機会を意味します。「不必要/無制限のアクセス」(値:0)、「容易なアクセス」(値:1)、「中程度のアクセス」(値:4)、「困難なアクセス」(値:10)など、攻撃が成功するまでに、攻撃が発見されないように製品にアクセスすることの困難さに応じて重み付けされています。
機器	攻撃に必要なソフトウェアやハードウェアを意味します。「標準機器」(値:0)、「特殊機器」(値:4)、「特別注文機器」(値:7)など、機器の入手の困難さに応じて重み付けされています。

攻撃能力についての詳細は、CEM[1]「附属書 B.4 攻撃能力の計算」を参照してください。

3.2 脆弱性評価の合否判定

CC では、製品が攻撃されたとしても耐えなければならない攻撃能力の値が、EAL 毎に決められています。例えば、EAL1～EAL3 の場合は「基本」(値:0～9)」、EAL4 の場合は「強化基本」(値:10～13)」です。

CC の脆弱性評価の合否は、CC 評価で検出された脆弱性を悪用するために必要な攻撃能力と、EAL 毎に決められた基準値を比較して判定されます。

■ 脆弱性の悪用に必要な攻撃能力が基準値を超えている場合
攻撃者が脆弱性を悪用して攻撃を成功させることは困難であることを意味します。製品にそのような脆弱性が検出された場合、製品が耐えなければならない基準値はクリアしているので､製品に脆弱性が存在していても CC 評価は合格となります。

■ 脆弱性の悪用に必要な攻撃能力が基準値を下回る場合
攻撃が容易に成功することを意味します。製品にそのような脆弱性が検出された場合、製品が耐えなければならない基準値をクリアしていないので、CC 評価は不合格となります。

3.3 攻撃能力算出と合否判定の例

攻撃能力の算出と脆弱性評価の合否判定について、具体的な例をあげて説明します。

■ 例 1
IT 技術に精通している専門家が、インターネットで入手できる攻撃ツールを改造して適用すると、2 週間以内に攻撃が成功する。

この場合の攻撃能力の計算例を表 3-2 に示します。なお、例 1 の中で、明示的に記述されていない要因は、攻撃される製品にとって最も不利な条件を適用しています。

表 3-2　攻撃能力の計算例（ツール改造の場合）

要因		値
所要時間	～2 週間	2
専門知識	エキスパート	6
評価対象の知識	公開	0
機会	無制限のアクセス	0
機器	特殊（改造したツール）	4
合計（攻撃能力）		12

表 3-2 から、この脆弱性を悪用するために必要な攻撃能力は 12（＝「強化基本」）となり、脆弱性評価の合否判定は次のようになります。

・EAL1～EAL3 の場合（「基本」攻撃に対する耐性が必要）：合格
・EAL4 の場合（「強化基本」攻撃に対する耐性が必要）　：不合格

■ 例 2
経験者が、インターネットで公開されている攻撃ツールを入手して適用すると、簡単に攻撃が成功する。

この場合の攻撃能力の計算例を表 3-3 に示します。なお、例 2 の中で、明示的に記述されていない要因は、攻撃される製品にとって最も不利な条件を適用しています。

表 3-3　攻撃能力の計算例（公開ツール利用の場合）

要因		値
所要時間	～1 日	0
専門知識	熟練者	3
評価対象の知識	公開	0
機会	無制限のアクセス	0
機器	標準（インターネットで入手したツール）	0
合計（攻撃能力）		3

表 3-3 から、この脆弱性を悪用するために必要な攻撃能力は 3（＝「基本」）となり、脆弱性評価の合否判定は次のようになります。

・EAL1～EAL3 の場合（「基本」攻撃に対する耐性が必要）：不合格
・EAL4 の場合（「強化基本」攻撃に対する耐性が必要）　：不合格

3.4 攻撃能力算出の注意点

攻撃能力の算出にあたっては、以下のような注意が必要です。

■攻撃のための情報やツールの入手可能性の考慮

前節の例 1 と例 2 の違いは、製品にそのまま適用できる攻撃ツールが、インターネットで入手できるか否かの違いです。このように、攻撃ツールをはじめ、脆弱性を悪用するための具体的な方法や手順などの情報がインターネットで入手できるか

どうかによって、そのツール自体が高度な手法を用いるかどうかに係らず、攻撃を成功させるための困難さ（攻撃能力）は大きく異なります。したがって、評価者は、攻撃のための情報やツールが公開されていないかどうか、インターネット等を十分に探索した上で、攻撃能力を計算することが必要です。

■1 つの脆弱性に対する複数の攻撃シナリオの考慮

1 つの脆弱性に対して、攻撃方法は 1 つとは限りません。多くの場合、いくつかの攻撃方法が存在します。例えば、パスワードの解析や製品特有の脆弱性の攻撃の場合を例にとると、次のようにいくつかの攻撃シナリオが考えられます。

・パスワードの解析
- しろうとが、総当たりでパスワードを入力し、時間をかけて解析する。
- 熟練者が、数多くの PC を利用して、短時間で解析する。

・製品特有の脆弱性の攻撃
- エキスパートが、公開情報をたよりにリバースエンジニアリングを行い、時間をかけて攻撃コードを開発する。
- エキスパートが、秘密情報を入手し、短時間で攻撃コードを開発する。

攻撃シナリオによって、攻撃能力の要因毎の重みが異なり、攻撃能力の値が変化します。したがって、評価者は、脆弱性を悪用するために考えられる攻撃シナリオを洗い出した上で、各攻撃シナリオを実行するために必要な攻撃能力を計算し、すべての攻撃シナリオに対して評価対象製品が所定の耐性を持っているかどうかを判定することが必要です。

■最新情報の考慮

脆弱性の悪用に必要な攻撃能力は、時とともに変化していく可能性があります。例えば、「ハードウェアの性能向上により攻撃の所要時間が短縮された」、「新たな攻撃ツールが入手可能になった」という場合には、必要な攻撃能力は小さくなります。したがって、評価者は、過去に類似の評価をした経験があったとしても、その過去の経験に基づいて評価をするのではなく、評価実施時点の最新の情報に基づいて、評価をすることが必要です。

4 脆弱性探索

本章では、評価対象製品に存在する可能性のある脆弱性を識別するために、CC で求められている脆弱性探索について説明します。

4.1 脆弱性探索手法

CC 評価の最終目的は、評価対象製品に悪用可能な脆弱性がないことを保証することです。したがって、評価者が製品に存在する可能性のある脆弱性を探索する際には、可能な限り考慮漏れをなくすことが重要です。そのために、CC では次のような脆弱性探索手法が採用されています。

■公知の脆弱性探索

CC では、評価者は、製品分野や評価者の関心の分野等に基づいて、一般に知られている脆弱性（これを「公知の脆弱性」といいます。）を探索することが求められています。これにより、脆弱性の考慮漏れを防止します。

■証拠資料分析による脆弱性探索

CC では、評価者は、一般的な脆弱性の観点を考慮して評価対象製品の設計情報等を分析し、脆弱性の可能性を仮定することが求められています。これにより、一般に知られている脆弱性だけでなく、製品固有の設計や実装に依存する脆弱性も検出することができます。

■欠陥仮説法による脆弱性探索（EAL4 以上の場合）

CC では、EAL4 以上の場合、評価者は、一般に知られている脆弱性探索手法である「欠陥仮説法（Flaw Hypothesis Methodology）」[3]を使用することが求められています。証拠資料の分析の際に、欠陥仮説法を適用することで、考慮漏れをさらに低減することが期待されます。

各脆弱性探索手法について、以下に詳細を説明します。

4.2 公知の脆弱性探索

公知の脆弱性を探索する際の情報源や探索すべき情報について、ヒントや注意点を説明します。

(1) 公知の脆弱性の情報源

公知の脆弱性は、インターネットで公開されている各種情報や書籍などを探索することによって得られます。代表的な情報源には次のようなものがあります。

■インターネット全般
・検索エンジンによる検索

■製品毎の発見された脆弱性
・JVN iPedia(脆弱性対策情報データベース)[4]
・CVE(Common Vulnerabilities and Exposures)[5]
・その他

■製品に依存しない一般的な脆弱性
・CWE(Common Weakness Enumeration)[6]
・IPA: 安全なウェブサイトの作り方[7]
・IPA: セキュア・プログラミング講座[8]
・その他

■攻撃の観点での情報
・Exploit Database[9]
・その他

(2) 公知の脆弱性の探索情報

公知の脆弱性の情報源を探索する際に考慮すべき観点やキーワードには次のようなものがあります。

■製品分野

製品分野に関連するキーワードを探索すると、その製品分野に特有の機能の脆弱性や、製品分野に発生しやすい脆弱性の情報が得られることが期待されます。そのための探索キーワードとしては、以下のようなものが考えられます。

・製品種別
ファイアウォール、DBMS、デジタル複合機などが該当します。

・類似製品
評価対象製品の旧製品や製品のシリーズ名、競合他社の製品も探索候補となります。また、製品ベンダー自身が提供している、製品のセキュリティ情報やソフトウェアの修正情報も該当します。

・派生元製品

製品の中には、別の製品をそのまま、あるいは、カスタマイズして、組み込んでいる場合があります。そのような派生元の製品名称も探索キーワードとなります。例えば、評価対象製品がSSL/TLS機能を実装しており、実装のために「openssl」が使われている場合、「openssl」の脆弱性を探索します。

■採用技術

製品で採用されている技術分野を探索すると、その技術分野で発生しやすい脆弱性情報が得られることが期待されます。例えば、セキュリティ技術、ネットワークプロトコル、製品内部で採用しているコンポーネント、プログラムの実行環境や実装技術、製品が扱うデータ形式などについて、具体的な技術名称や製品名称を探索します。

■関心の分野

評価者が関心を抱く分野には、例えば、次のようなものがあります。

・脆弱性が数多く知られている分野（Webインタフェース、入力処理等）
・製品独自の仕様、機能、インタフェース
・複雑な仕様や機能、他

(3) 公知の脆弱性探索の注意点

公知の脆弱性を探索する際には、次のような注意が必要です。

■別製品の脆弱性の考慮

公知の脆弱性探索の目的は、考慮漏れを少なくすることです。したがって、公知の脆弱性探索で評価対象製品以外の製品の脆弱性情報が得られた場合、「別製品なので評価対象製品には該当しない」と考えるのではなく、「別製品で発見されたのであれば、当該製品にも該当するかもしれない」と考えて、存在する可能性のある脆弱性を仮定します。

■情報源のかたよりの排除

インターネット全般を探索すると、脆弱性に関する論文やカンファレンスでの発表資料を含めて、脆弱性に関する多くの情報を得ることができます。それに対して、CVE[5]などの脆弱性情報に特化したサイトを検索すると、効率よく脆弱性情報を得ることができます。ただし、特定のサイトだけに依存すると、他の情報源を検索すれば容易に得られる脆弱性情報が発見されない可能性があるので、注

意が必要です。

■探索の繰り返し

ある検索結果で得られた情報から、さらに関連する情報を検索するといったように、検索を繰り返す必要が生じる場合もあります。例えば、製品毎の脆弱性情報の多くは、脆弱性の概要だけを公開しており、詳細な情報は公開していません。その場合に、製品名や脆弱性のキーワード等を元に、攻撃の観点での情報源や脆弱性に関する各種フォーラム等を検索すると、追加の情報が得られることがあります。

4.3 証拠資料分析による脆弱性探索

製品の設計書や利用者向けガイダンスなどの証拠資料を分析して脆弱性を仮定する際の、ヒントや注意点を説明します。

(1) 脆弱性の観点

CC では、評価者は、表 4-1 に示す一般的な脆弱性の観点を考慮して脆弱性を探索することが求められています。

表 4-1　一般的な脆弱性の観点

	カテゴリ	内容
a)	製品種別に関する一般的脆弱性	公知の脆弱性探索で得られる一般的な脆弱性です。
b)	バイパス	攻撃者がセキュリティ機能の適用を回避できる場合が該当します。機密データが読み出される場合も含まれています。
c)	改ざん	攻撃者がセキュリティ機能の実行コードやデータを改変することにより、本来意図されていない処理を実行したり、セキュリティ機能を停止したりする場合が該当します。
d)	直接攻撃	攻撃者がパスワードによる認証等のメカニズムに対して、試行を繰り返すなど、直接攻撃する場合が該当します。
e)	監視	攻撃者が製品の動作や送受信データを監視することにより、製品が保護する機密情報を入手する場合が該当します。
f)	誤使用	製品の利用者向けガイダンスが、適切に記述されていなかったり、セキュリティを維持するために大きな負担のかかる運用管理を求めたりすることで、利用者が製品のセキュアな管理や使用ができない場合が該当します。

各カテゴリ内容の詳細は、CEM[1]「附属書 B.2.1 一般的な脆弱性に関するガイダンス」を参照してください。

また、EAL2 以上の場合には、評価者は、開発者が証拠資料として提示する「セキュリティアーキテクチャ記述」を考慮することが求められています。セキュリティアーキテクチャとは、製品のセキュリティ機能がバイパスや改ざんされないように保護するメカニズムのことです。その内容は、脆弱性探索のために重要な情報となります。セキュリティアーキテクチャについての詳細は以下を参照してください。

・IPA: 開発者のためのセキュリティアーキテクチャ解説[2]

(2) 脆弱性の観点に基づいた探索手順例

CC では、評価者は前節の脆弱性の観点を考慮することが求められていますが、具体的な方法は規定されていません。ここでは、本書の読者の理解のために、証拠資料を分析し脆弱性を探索する手順の一例を紹介します。

本探索手順では、製品のセキュリティ機能やインタフェースの１つ１つについて、表 4-1 の脆弱性の観点で証拠資料を分析し、製品に存在する可能性のある脆弱性を探索します。その場合の脆弱性探索の流れは以下のとおりです。

①製品のセキュリティ機能とインタフェースの特定

評価者は、製品の設計書や利用者向けガイダンスなどの証拠資料を分析して、製品のセキュリティ機能やインタフェースをリストアップします。

セキュリティ機能に関係がないように思われるインタフェースであっても、例えばバッファオーバフローのような、セキュリティ機能を侵害する脆弱性が存在する可能性があります。そのような脆弱性も検出できるように、すべてのインタフェースを対象とします。

②脆弱性の仮定

評価者は、リストアップしたセキュリティ機能やインタフェースの各々に対して、表 4-1 の観点を適用し脆弱性を仮定します。

例えば、パスワードによる認証機能に対して、表 4-1 の観点を適用すると、表 4-2 に示す脆弱性の可能性が考えられます。

表 4-2　脆弱性の観点の適用例

	カテゴリ	脆弱性の例
a)	製品種別に関する一般的脆弱性	公知の脆弱性で得られた「Rainbow Attack」などの攻撃手法が成功する懸念が考えられます。
b)	バイパス	Web システムの場合、URL を直接指定することで、認証なしでアクセスが成功する懸念が考えられます。 また、パスワードの格納場所から何らかの方法でパスワードを取得し、認証に成功する懸念が考えられます。
c)	改ざん	不正なデータを入力することによってバッファオーバフローや SQL インジェクションの攻撃が成功し、本来意図していない処理が実行される懸念が考えられます。 また、パスワードを格納したファイルを何らかの方法で変更または上書きし、認証に成功する懸念が考えられます。
d)	直接攻撃	様々なパスワードを繰り返し試行すると、認証が成功する懸念が考えられます。
e)	監視	ネットワークに送信されたパスワードが盗聴される懸念が考えられます。
f)	誤使用	製品の機能や利用者向けガイダンスが適切でないために、パスワード試行回数の制限など、認証機能を補強するための機能が働かない設定になっていることに気が付かないまま運用される懸念が考えられます。

③証拠資料の分析

評価者は、証拠資料を検査して、仮定した脆弱性があてはまるかどうかを分析します。証拠資料にセキュリティアーキテクチャ記述が含まれている場合には、その保護メカニズムも考慮します。仮定した脆弱性は、以下のいずれかに該当します。

■脆弱性に対する対策が判断できない場合

仮定した脆弱性が製品に存在する可能性を否定できません。脆弱性の有無は、2 章で説明したように、脆弱性を悪用するために必要な攻撃能力も考慮した上で、侵入テストで確認することになります。

■脆弱性に対する対策がされている場合

例えば、「監視」に対してネットワーク通信の暗号化による対策が実施されている場合が該当します。この場合には、評価者は、評価者または開発者が実施

したテスト結果の資料を検査して、製品の設計書に記述されている対策の動作（本例では、ネットワーク通信の暗号化によるパスワードの保護）がテストされているかどうかを確認します。テストが充分でない場合には、評価者は動作確認テストを実施して、対策の動作を検証します。

これらの確認によって、分析対象の脆弱性（本例では、監視）に対しては、対策がされているため、脆弱性は存在しないという結論になります。しかし、脆弱性の対策（本例では、ネットワーク通信の暗号化）については、さらに、次の④の分析が必要となります。

④脆弱性対策に対する分析

仮定した脆弱性に対して対策が実施されていたとしても、その対策の実装の仕方等に問題があれば、攻撃者はその問題を悪用して製品のセキュリティ機能を侵害する可能性があります。そのため評価者は、製品の脆弱性対策に対しても、その対策の不備や弱点を探索するために、②と③の分析を行います。例えば、監視に対してネットワーク通信の暗号化による対策が実施されている場合、ネットワーク通信の暗号化について②と③の分析をすることになります。

セキュリティアーキテクチャ記述に記載されている保護メカニズムは、本分析の対象となります。

⑤すべてのセキュリティ機能やインタフェースの分析

評価者は、製品のすべてのセキュリティ機能やインタフェースについて、②～④の分析を実施します。

(3) 証拠資料分析の注意点

証拠資料を分析し脆弱性を探索する際には、次のような注意が必要です。

(3-1) ソースコードの分析

評価者は、製品のソースコードを参照することによって、設計書に記述されていないような実装レベルを含めて、より詳細な分析ができることになります。ソースコードならではの脆弱性探索の観点には次のようなものがあります。

■セキュリティ機能が依存しているデータや処理の詳細分析

評価者は、ソースコードの中で、セキュリティ機能が参照したり処理したりしているデータや、セキュリティ機能内部での分岐処理等に着目します。評価者は、

それらのデータに対して異常・不正な値を設定したり、分岐条件を誤らせたりするような使い方ができないかどうかを分析します。

■コーディング上の公知の脆弱性に着目した分析

評価者は、ソースコードを分析する際に、脆弱性の要因となるコーディング上の問題を考慮します。コーディングに起因する一般的な脆弱性は、4.2 節で説明した公知の脆弱性検索で得られます。例えば、以下のような問題が知られています。

・バッファオーバフロー（スタック、ヒープ）
・整数オーバフロー
・メモリの不正使用（解放したメモリの使用や、メモリの二重解放など）
・競合、他

■コンパイラに着目した分析

CC では、ソースコードを扱う EAL4 以上の場合、評価者は、プログラミング言語の処理系依存の構文やコンパイラの仕様を評価した上で、ソースコードにコンパイラに起因する脆弱性がないかどうかを分析します。

開発者のコーディングの仕方によっては、コンパイラによって開発者の意図と異なるコードが生成され、脆弱性がもたらされる可能性があります。例えば C 言語の場合、以下のような問題が知られています。

・最適化
　コンパイラの最適化処理によって、処理の順序が変更されたり、冗長な処理が削除されたりする場合があります。開発者が、ハードウェアを制御するコードを記述したり、セキュリティ上の要請から故意に冗長なコードを記述したりしている場合には、悪影響が発生する場合があります。

・char 型の符号
　C 言語の規格では、signed／unsigned が明示的に指定されていない char 型の符号は、処理系依存とされています。開発者の想定している char 型の符号とコンパイラの char 型の符号が異なる場合、127 を超える値の比較や加減算において、開発者の意図と異なる結果となります。これにより、char 型変数の使われ方によっては、セキュリティ機能に悪影響が発生する場合があります。

■隠し機能や隠しオプション

製品によっては、設計書や利用者向けガイダンスに記載されていない機能やオプション指定を備えている場合があります。また、製品開発時のデバッグ用の機

能が誤って最終製品に残存している場合もあります。それらの機能は、セキュリティ上の問題がないかどうか開発者内での充分な検討やテストがされず、脆弱性が存在する可能性があります。

なお、CC では、評価者は、脆弱性評価の前に、製品の仕様書や利用者向けガイダンスと照らし合わせてソースコードを分析し、製品の機能が正確に実装されているかどうかを評価します。多くの場合、その評価の中で、隠し機能や隠しオプションは検出されます。

■複雑な処理

ソースコードによっては、処理が複雑でわかりにくい場合があります。評価者は、複雑な処理を発見した場合、ソースコード分析だけでは検出しにくい脆弱性の存在を仮定して、テストによって脆弱性の有無を確認することがあります。

(3-2) 脆弱性可能性の判定

証拠資料の分析では、評価者は、脆弱性の可能性を新たに検出することに加えて、製品にあてはまるかどうか不確かな脆弱性が発生する可能性がないことを決定します。その際に、脆弱性の対策がされているからと言って、安易に問題がないと決定しないように十分な注意が必要です。脆弱性対策の設計上の考慮漏れや、コーディングミスによって、攻撃に悪用される余地が残されている場合があります。

例えば、製品が、入力された文字列の中の特殊文字やキーワード等をチェックして悪影響がないように処理している場合、以下に示すような問題が懸念されます。

■チェック漏れ

設計書やプログラムで、チェックすべき文字が漏れている場合が考えられます。一般に、ブラックリスト方式は抜け漏れが発生しやすいため、注意が必要です。

■エンコーディングの考慮漏れ

文字は、16 進表記、%表記、UTF-7/8 など、様々な形式で表現可能な場合があります。設計書やプログラムで、その考慮が漏れている場合が考えられます。

■チェック時と文字列が使用される時の解釈の違いの考慮漏れ

文字列のチェック時と文字列が実際に使用される時で、特殊文字の解釈が異なる場合があります。例えば、文字列の途中の NULL 文字や改行コードは、文字列のチェック時には 1 つの文字としてみなされ、文字列が実際に使用される時には無視される可能性があります。設計書やプログラムで、その考慮が漏れている場合が

考えられます。評価対象製品が、Web ブラウザや SQL を処理する DBMS のような他の IT 製品にデータを渡す際には、このような解釈の違いに充分な注意が必要です。

4.4 欠陥仮説法による脆弱性探索

CC の EAL4 以上の評価では、証拠資料を分析して脆弱性を探索する際に、欠陥仮説法を使用することが求められています。ここでは、欠陥仮説法の概要を説明します。

(1) 欠陥仮説法の概要

欠陥仮説法は一般に知られている脆弱性探索手法です。CC には欠陥仮説法の説明は記述されていません。欠陥仮説法（Flaw Hypothesis Methodology）の詳細については、次の参考文献を参照してください。

・Clark Weissman,” Penetration Testing”[3]

欠陥仮説法は次の 4 つの段階で構成されています。

①欠陥の仮定（Flaw generation）
疑われる欠陥の仮説を生成します。

②欠陥の検証（Flaw confirmation）
欠陥の仮説が正しいかどうかを検証します。

③欠陥の一般化（Flaw generalization）
確認された欠陥をもたらす一般的な弱点を分析します。

④欠陥の除去（Flaw elimination）
確認された欠陥を報告します。（開発者は何らかの対処をします。）

欠陥仮説法による証拠資料分析の流れを図 4-1 に示します。

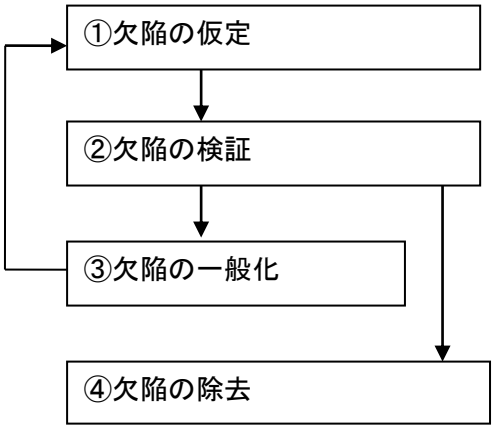

図 4-1　欠陥仮説法による分析の流れ

(2) 欠陥仮説法の内容

欠陥仮説法の 4 つの段階について説明します。

①欠陥の仮定

評価者は、製品の設計情報（製品が実現すべきセキュリティポリシー、外部インタフェース仕様、製品内部の設計書、ソースコード)、利用者向けガイダンスを分析して、製品の欠陥や運用上の欠陥を仮定します。欠陥を仮定する際の観点には、例えば、次のようなものがあります。

・発見された欠陥と類似の欠陥
・製品が実現すべきセキュリティポリシーと、設計や実装との乖離
・製品の設計と実装との乖離
・製品のアーキテクチャや機能の不備
・実装ミス
・開発作業に起因する欠陥（例えば、デバッグ機能の残存など）
・問題を引き起こすような利用方法

これらの具体的な実施方法は、「4.2 公知の脆弱性探索」や「4.3 証拠資料分析による脆弱性探索」で説明した内容とほぼ同じです。

②欠陥の検証

評価者は、仮定した欠陥が評価対象製品にあてはまるかどうかを、ソースコードや実施済のテスト結果等を分析して決定します。

また、評価者は、分析途中の仮説を確認するために、テストを実施する場合もあります。この場合のテストは、最終的に脆弱性が悪用可能であるかどうかを判定するための侵入テストとは目的が異なり、仕様の確認のためのテストも含まれています。評価者は、仮説の検証結果を基に、仮説をさらにすすめたり、仮説を変更したりして、欠陥の可能性を探索していきます。評価者は、テストを実施しながら分析をすすめることにより、評価者の製品に対する理解を深めるとともに、ソースコードの詳細分析よりもすばやく仮説を検証することができます。

③欠陥の一般化

評価者は、確認された欠陥を分析して「欠陥の一般化」を行い、その一般的な欠陥を①の分析の入力として、新しい仮説を立案します。欠陥の一般化には次のような考え方が含まれています。

■欠陥の原因となるメカニズムの特定

評価者は、確認された欠陥を分析して、その原因となるメカニズム（弱点）を特定します。その弱点は、別の処理にもあてはまる可能性があります。

例えば、製品のアクセス制御機能において、アクセス制御機能をバイパスすることのできる欠陥が確認されたとします。それを「アクセス制御機能のバイパス」として捉えるのではなく、さらに、その原因となるメカニズムを分析します。

例えば、アクセス制御機能のバイパスの原因が、アクセス権限のチェックと実際にアクセスが行われる間の時間差の悪用（Time-of-check to Time-of-use）だったとします。その場合、一般化された弱点は、「Time-of-check to Time-of-use」の問題となり、アクセス制御機能だけでなく、データの妥当性を検証している処理全般にあてはまる可能性があります。

■欠陥の原因となるメカニズムの一般化

評価者は、さらに、欠陥の原因となるメカニズム（弱点）について、それをより一般化した上位の概念の弱点を考察します。

例えば、「Time-of-check to Time-of-use」（CWE[6]の CWE-367 に該当）の上位の弱点として「Race Condition」（CWE[6]の CWE-362 に該当）が考えられます。そこで、より一般化した「Race Condition」という観点で製品の欠陥を探索すると、共有データに関する競合など別の欠陥が発見される可能性があります。

■欠陥の結合

製品に複数の欠陥があり、個々の欠陥はセキュリティにそれほど深刻な影響をもたらさない場合であっても、複数の欠陥を組み合わせると深刻な影響をもたらす可能性があります。

例えば、製品において次の 2 つの欠陥（弱点）が発見されたとします。

・インタフェース A
　製品が保護している重要なデータにアクセスできる可能性があります。しかし、そのためには設定ファイルを変更することが必要となりますが、設定ファイルを変更する手段は提供されていません。

・インタフェース B
　製品内のいくつかのファイルを変更できる可能性があります。しかし、製品が保護している重要なデータにはアクセスすることはできません。

この場合、評価者がインタフェース A とインタフェース B を別々に分析すると、製品のセキュリティに悪影響を与えることのできる脆弱性は存在しないという結論が導かれる可能性があります。

しかし、2 つのインタフェースを組み合わせると、「インタフェース B を利用してインタフェース A の設定ファイルを変更すると、製品が保護している重要なデータにアクセスできるかもしれない」という仮説が考えられます。

④欠陥の除去

評価者は、発見された製品の欠陥と推奨措置を、開発者に報告します。開発者は、欠陥の内容を分析し、製品の修正や回避策を検討し、欠陥を取り除きます。

4.5 脆弱性に関する詳細情報の探索

公知の脆弱性探索は、製品に存在する可能性のある一般的な脆弱性を識別する目的に加えて、製品に存在することが仮定された脆弱性についてさらに詳細な情報を得る目的にも使用されます。後者の場合の公知の情報探索について、以下に説明します。

(1) 脆弱性が悪用可能か否かの判定

3 章の「攻撃能力」で説明したように、攻撃方法が知られているかどうかは、脆弱性が悪用可能かどうかの判定に大きく影響します。特に、攻撃コードやツールが公開されている場合には、技術的には困難に思われるような攻撃であっても、攻撃者は比較的簡単に実践することができてしまいます。また、それらの攻撃のための情

報は、評価者が侵入テストを実施するためにも必要となります。

そのため、公知の脆弱性探索や証拠資料探索で発見された脆弱性に対しては、具体的に攻撃するための情報が公開されていないかどうか、公知の情報を充分に探索することが必要となります。

(2) 開発者の主張や対策の反証

評価者は、製品がセキュリティのために何らかの対策を実施していた場合、「対策されているから問題ない」という発想ではなく、攻撃者の視点でその対策に脆弱性がないかどうかを分析することが求められます。そのようなセキュリティ対策の反証の際にも、公知の情報を充分に探索することが必要となります。

セキュリティ対策の反証のために、公知の脆弱性探索を繰り返し実施する場合の例を以下に示します。

■例：暗号鍵の漏えい対策の反証

評価対象製品が暗号化機能を提供しており、証拠資料では暗号鍵を取り出す手段を提供していない場合を考えます。それに対して評価者は、攻撃者と同じ視点で、その反証を試みます。

①暗号鍵を読み出す手段は本当にないのか？

インターネットを検索すると、読み出し方法のヒントが得られます。

- OS のデバイスファイル（/dev/mem 等）
- プロセスのコアダンプファイル
- 二次記憶装置（ページングやスワップのための領域）
- DRAM からの読み出し

ただし、仮にメモリ内容を読み出すことができたとしても、メモリに記録されている大量のデータの中から、暗号鍵を特定し取り出すことは困難であるように思われます。

②メモリ中の暗号鍵を特定することは本当に困難か？

インターネットを検索すると、メモリ中の暗号鍵を特定するアルゴリズムに関する論文が得られます。

Lest We Remember: Cold Boot Attacks on Encryption Keys, Proc. 17th USENIX Security Symposium, 2008

ただし、論理的には可能であっても、実践することは困難かもしれません。

③論文の内容は本当に実践可能か？

さらにインターネットを検索すると、論文の内容を実証するプログラム「*aeskeyfind*」が存在することがわかります。

以上のように、評価者は、攻撃の障壁が存在したとしても、その障壁を打ち負かすことはできないか？という観点で探索を行っていきます。評価者は、充分な探索を行った後に、最も簡単に実行できる攻撃シナリオを考え、その攻撃能力を計算し、必要に応じて侵入テストを実施することになります。

5 侵入テスト

本章では、評価対象製品に存在する可能性のある脆弱性が、実際に存在するかどうかを確認するための侵入テストについて説明します。

5.1 侵入テストの概要

脆弱性分析の結果、評価対象製品に存在する可能性のある脆弱性がリストアップされます。評価者は、それらの脆弱性が本当に評価対象製品に存在するかどうか、侵入テストを実施して判定します。

侵入テストの実施にあたっては、目的の脆弱性に対して様々な攻撃の可能性を考慮して、適切なテスト方法を考案する必要があります。CC では、侵入テストの具体的方法については、規定していません。インターネットを検索すると、侵入テスト方法の解説や、侵入テストツールなどの情報を得ることができます。以下に、参考文献の例を示します。

・オライリー・ジャパン:
　実践 Metasploit - ペネトレーションテストによる脆弱性評価[10]
・OWASP Testing Project: OWASP Testing Guide[12]

5.2 侵入テストの注意点

侵入テストを考案したり実施したりする際の、ヒントや注意点について説明します。

(1) 攻撃のバリエーションの考慮

目的の脆弱性に対して、攻撃のバリエーションは多数存在する可能性があります。そのため、開発者が目的の脆弱性に対して対策をしていても、対策をすり抜ける攻撃が存在することが懸念されます。評価者は、公知の脆弱性を探索して、これらの攻撃のバリエーションを充分に考慮して、侵入テストを実施する必要があります。

例えば、クロスサイトスクリプティングの場合、文字コードのエンコーディング、利用する HTML のタグなど、数多くのバリエーションが知られています。詳細は以下を参照してください。

・OWASP: XSS Filter Evasion Cheat Sheet[13]

(2) 検査ツールの活用

多数の攻撃のバリエーションなどを検査する際には、それに対応した検査ツールを利用すると、効率的に検査することができます。

インターネットを検索すると、多数の検査ツールが存在します。検査ツールを選定する際には、評価者が検査したい脆弱性が、その検査ツールで十分に検査できるかどうかに注意する必要があります。検査ツールで、評価者の目的とするような検査ができない場合には、評価者は自身で検査方法を考案する必要があります。

検査ツールの中には、検査のために必要となる共通的な機能をフレームワークとして提供し、具体的な検査データ（攻撃コード）は必要に応じて追加できる構造のものがあります。そのような検査ツール用の検査データ（攻撃コード）をインターネットで検索すると、目的の脆弱性用の検査データ（攻撃コード）を得ることができる場合があります。

(3) 攻撃の成否の判定

侵入テストの内容を策定するにあたっては、評価対象製品に入力する検査データ（攻撃データ）だけでなく、実施したテスト（攻撃）が成功したか失敗したかの判定にも注意が必要です。場合によっては、製品の応答では成功失敗が判定しにくいために、攻撃が成功しているにもかかわらず、攻撃の成功を見落とす可能性もあります。

例えば、SQL インジェクションの場合を考えます。多くの解説でわかり易い事例として使われているログインの成功などは、極めてまれな事例です。実際には、多くの場合、単なるエラーが表示されるなど、SQL インジェクションの成功失敗が判別しにくい結果となります。エラーが表示されたとしても、製品内部で SQL が実行されている可能性があるため、注意が必要です。

攻撃の成功失敗の判定が困難な場合、開発者の協力が得られるときには、製品内部のログ等によって攻撃の成功失敗を判断できる場合があります。それができない場合には、評価者は攻撃者と同じ視点で攻撃の成功失敗を判断することになります。例えば、時間のかかるコマンドを実行し、成功時と失敗時の応答時間の違いを観測するなどの方法があります。

SQL インジェクションの場合には、「ブラインド SQL インジェクション」と呼ばれる攻撃手法が知られています。その検査方法の詳細については、「OWASP Testing Guide」[12]を参照してください。

(4) ポートスキャン検査

一般に、TCP/IP のオープンポートを調査するために、ポートスキャン用のツールを用いた検査が行われています。ポートスキャンの検査の際には、次のような注意が必要です。

■動的なオープンポートの考慮

ポートスキャンを実施するにあたっては、実施するタイミングにも注意が必要です。評価対象製品の機能によっては、製品の利用に伴って、動作途中でポートがオープンされる場合があります。そのため、ポートスキャンを実施するタイミングによっては、それらの動的にオープンされるポートが検出できない可能性があります。

■想定外のオープンポートが検出された場合の検査

CC では、開発者は、評価対象製品の外部インタフェース仕様を提供し、それに基づいて評価が行われます。したがって、評価対象製品のオープンポートは、検査前に決まっています。それにもかかわらず、想定外のオープンポートが検出された場合には、脆弱性以前に、外部インタフェース仕様として提出された証拠資料の不備が疑われることになります。

■正当なオープンポートが検出された場合の検査

外部インタフェース仕様どおりのオープンポートが検出された場合、仕様どおりのインタフェースであるという理由だけで、脆弱性がないと判断することはできません。次に説明するように、ネットワークプロトコルに対する脆弱性の可能性を追跡検査する必要があります。

(5) ネットワークプロトコルの検査

ネットワークプロトコルは、仕様そのものや設定ミスによって、攻撃に悪用される可能性のある機能が含まれている場合があります。

例えば、FTP サーバの場合、対象ファイルのアクセス権限の設定ミス、CD コマンドによる想定外のディレクトリ参照、SITE EXEC コマンドによる想定外のプログラム実行、PORT コマンドの悪用（FTP バウンス攻撃）などが該当します。また、製品にメインテナンス用のアカウントが設定されている場合、そのアカウントを使用した不正ログインも懸念されます。

評価者は、脆弱性探索においてそのようなプロトコルの弱点を調査しておき、侵入テストを実施して、悪用可能性がないことを確認します。

(6) 実装上の欠陥の検査

製品には、バッファオーバフローをはじめとする実装上の欠陥による脆弱性が存在する可能性があります。製品のソースコードを参照することができない検査においては、実装上の欠陥を仮定し、テストによって確認することになります。

そのような明確になっていない脆弱性をテストする手法として、「ファジング」という手法が知られています。例えば、ネットワークプロトコルのコマンドやデータ形式等の各種フィールドに対して、バッファオーバフローを引き起こす可能性のあるデータ、誤動作を発生する可能性のある特殊文字などを含むデータを評価対象製品に入力し、その反応を確認します。それによって、脆弱性が存在しないかどうかを検査します。ファジングについては、以下の参考文献を参照してください。

・IPA: ファジング活用の手引き[14]

6　おわりに

本書では、CC の脆弱性評価の概要について、実際に脆弱性の探索や侵入テストを実践する際の注意点を含めて説明しました。

CC では、均質な評価結果が得られるように評価方法が決められており、「ベストプラクティス」と考えられる評価方法が規定されています。ただし、その評価方法を実践するためには、インターネット等を充分に探索し、得られた情報を適切に活用することが必要です。

特にインターネットでは、脆弱性情報だけでなく、様々な攻撃ツールや攻撃コードが公開されています。攻撃ツール等が存在する場合、高度な知識や技術を必要とする攻撃であっても、比較的簡単に実行できてしまうため、大変危険です。脆弱性評価を実践する際には、それらの最新の情報に充分な注意が必要です。

また、近年、脆弱性が発見されてから攻撃コードの存在が知られるようになるまでの期間が短くなっており、いわゆる「ゼロデイ攻撃」が問題になっています。したがって、開発者は、製品に脆弱性が検出された場合、検出時点での脆弱性の悪用困難さだけでなく、悪用困難さの変化の可能性も考慮した上で、速やかに脆弱性の対処をすることが望まれます。

なお、脆弱性の内容を理解し侵入テストを考案するためには、IT 技術の基礎的な知識が必要です。例えば、以下のような文献が参考になります。

・オライリー・ジャパン:

Hacking: 美しき策謀 第 2 版—脆弱性攻撃の理論と実際[15]

参考文献

本書で参照している文献の一覧を以下に示します。URL は執筆時点のものです。

■全体

[1] *情報技術セキュリティ評価のための共通方法：評価方法，バージョン3.1 改訂第4版，2012 年 9 月，CCMB-2012-09-004，（平成 24 年 11 月，翻訳第 1.0 版）*
http://www.ipa.go.jp/security/jisec/cc/index.html
本書で解説している脆弱性の評価方法の規格書です。

[2] *IPA：開発者のためのセキュリティアーキテクチャ解説，*
http://www.ipa.go.jp/security/jisec/apdx.html#ADV_ARC_GUIDE
IT 製品のセキュリティ機能を攻撃から守るためのしくみであるセキュリティアーキテクチャについて解説しています。CC 評価では、開発者は、脆弱性評価が合格するように IT 製品を設計実装することが求められます。セキュリティアーキテクチャは、脆弱性対策に焦点をあてた設計実装内容に相当します。

■4 章 脆弱性探索

[3] *Clark Weissman,”Penetration Testing”, Information Security: An Integrated Collection of Essays, Essay 11, pp.269-296, IEEE Computer Society Press, 1995*
欠陥仮説法についての提唱者による解説です。

[4] *JVN iPedia（脆弱性対策情報データベース），http://jvndb.jvn.jp/*
日本国内で利用されているソフトウェア製品等の脆弱性対策情報です。

[5] *CVE（Common Vulnerabilities and Exposures），http://cve.mitre.org/*
個別製品に発見された脆弱性の情報です。登録情報には CVE 番号が付与されており、どの製品のどの脆弱性であるかを一意に識別することができます。

[6] *CWE（Common Weakness Enumeration），http://cwe.mitre.org/*
ソフトウェアの脆弱性（弱点）の種類の情報です。登録情報には CWE 番号が付与されており、ソフトウェアの脆弱性（弱点）の種類を一意に識別することができます。
CWE は、世の中で発見された脆弱性について、欠陥仮説法に含まれている「欠陥の一般化」を行い、導き出された共通的な弱点を階層構造に整理したものと考えられます。

[7] *IPA：安全なウェブサイトの作り方，*
http://www.ipa.go.jp/security/vuln/websecurity.html
Web アプリケーションの代表的な脆弱性とその対策方法を解説しています。

[8] *IPA: セキュア・プログラミング講座,*
http://www.ipa.go.jp/security/awareness/vendor/programming/
ソフトウェアの代表的な脆弱性について、設計、実装、テストといった開発工程全般での対策方法を解説しています。

[9] *Exploit Database, http://www.exploit-db.com/*
脆弱性やその攻撃方法に関する様々な情報が掲載されています。

■5章 侵入テスト

[10] *David Kennedy 他, 実践 Metasploit－ペネトレーションテストによる脆弱性評価, オライリー・ジャパン, 2012*
ペネトレーションテスト（侵入テスト）の一般的なやり方を、代表的ツール Metasploit を中心に解説しています。

[11] *Metasploit, http://www.metasploit.com/*
脆弱性検査ツールの１つである Metasploit の公式ホームページです。Metasploit 用の各種モジュールや、各種脆弱性を検査するための具体的な情報が掲載されています。

[12] *OWASP Testing Project, OWASP Testing Guide v3,*
https://www.owasp.org/index.php/OWASP_Testing_Project
・*原文: https://www.owasp.org/images/5/56/OWASP_Testing_Guide_v3.pdf*
・*日本語訳: https://www.owasp.org/images/1/1e/OTGv3Japanese.pdf*
Web アプリケーションの脆弱性を検査するための具体的な方法が記述されています。

[13] *OWASP: XSS Filter Evasion Cheat Sheet,*
https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
クロスサイトスクリプティングの攻撃に使用される可能性のある様々な表記方法が記述されています。

[14] *IPA: ファジング活用の手引き, http://www.ipa.go.jp/security/vuln/fuzzing.html*
脆弱性を検出する技術の 1 つであるファジングについて、概要と実践方法を解説しています。

■6章 おわりに

[15] *Jon Erickson, Hacking：美しき策謀 第２版－脆弱性攻撃の理論と実際, オライリー・ジャパン, 2011*
バッファオーバフロー、シェルコード、ネットワーク経由の攻撃、パスワードクラッキングなどの基本的な脆弱性や攻撃方法について、その技術的なメカニズムや実際の検証プログラムを解説しています。

開発者のためのセキュア解説（脆弱性評価編）

2013 年 3 月 4 日　　初版発行

著作・発行　　独立行政法人情報処理推進機構（IPA）

執　筆　者　　情報セキュリティ認証室