

SYSTEM DESIGN

開発者のためのセキュリティ入門

アンチパターンから学ぶ
セキュアシステム開発



ANTI-PATTERNS

改訂履歴

版数	改定年月日	改定箇所	改定内容
第 1 版	2024 年 7 月 31 日		初版発行

概要

この本はセキュリティ教育ドキュメントであり、読者には、システム開発者（主に設計開発者・品質保証担当者）として配属されたばかりの初学者を想定しています。この本を読むことで読者に「セキュリティ・バイ・デザインの意識」を学んでもらいたいと思っています。

セキュリティ・バイ・デザインとは、システムを設計する段階からセキュリティを考慮する思想です。端的に言えば、「セキュリティはやらなければならない。やらなければならないなら早い方がよい」ということです。システムを作る際、完成後にセキュリティ対策を後付けしようと思うと、完成前に組み込むよりも非常に大きい労力とコストがかかります。そこで、セキュリティ・バイ・デザインの意識を開発メンバや品質保証メンバに定着させれば、比較的少ない労力とコストで、システムのセキュリティレベルが向上できると考えました。

開発現場のセキュリティ対策を設計開発者・品質保証担当者向けに伝えるセキュリティ入門書は、一般層向けの入門書と比較すると、世の中に少ないように感じています。この本は、初学者に読みやすい工夫として、著者らで失敗例（アンチパターン）を考え、それを紹介する形式で執筆しました。アンチパターンは 12 個と、それほどたくさんある訳ではないので、幅広い方に読まれることを期待しています。

免責事項

- 本書は単に情報として提供され、内容は予告なしに変更される場合があります。
- 本書に記載の内容は、独立行政法人情報処理推進機構および産業サイバーセキュリティセンターの意見を代表するものではなく、筆者らの見解に基づいています。
- 本書の利用によるトラブルに対し、筆者らならびに監修者は一切の責任を負いません。
- 本書に誤りがないことの保証や、商品性又は特定目的への適合性の黙示的な保証や条件を含め明示的又は黙示的な保証や条件は一切ありません。
- 本書に登場する組織・企業名はすべて架空のものであり、実在の組織・企業とは一切関係ありません。
- 本書の有効期限は、第 1 版の発行日から 2 年間とします。

著作権およびその他すべての知的所有権

本書に関する著作権およびその他すべての知的所有権は、独立行政法人情報処理推進機構 産業サイバーセキュリティセンター 中核人材育成プログラム 7 期生 セキュリティ・バイ・デザイン「システム開発のセキュリティ向上 0.0」プロジェクトおよび本書中に利用した下記各イラスト制作者等に帰属します。

・いらすとや (<https://www.irasutoya.com/>)

また、本書に登場するシステム名、製品名は各社の商標または登録商標です。

目次

改訂履歴	1
概要	2
免責事項.....	2
著作権およびその他すべての知的所有権	2
目次	3
第 1 章 はじめに	4
1.1 背景.....	4
1.1.1 サイバーセキュリティの重要性.....	4
1.1.2 システムを作る側の責任	6
1.1.3 開発初期段階でのセキュリティ検討.....	7
1.2 本書について	14
1.2.1 本書の作成目的・対象読者.....	14
1.2.2 本書の特徴・スコープ・使い方.....	14
第 2 章 アンチパターン	16
2.1 アンチパターンの見方.....	16
2.2 アンチパターン集.....	18
第 3 章 おわりに	67
謝辞.....	67
付録 A セキュリティの学習方法.....	68
付録 B セキュリティ文化の形成.....	71

第1章 はじめに

1.1 背景

ここでは、本書作成の背景を説明します。

1.1.1 サイバーセキュリティの重要性

サイバー攻撃は増加している

近年、サイバー攻撃の件数は増えており、また情報技術の進化とともにサイバー攻撃の手法も多様化・巧妙化しています。

- ✓ 総務省の情報通信白書ⁱでは、観測されたサイバー攻撃関連通信数が 2015 年から 2022 年で 8.3 倍に増加したことが報告され、国民の誰もがサイバー攻撃の懸念に直面していると述べています。
- ✓ IPA の情報セキュリティ白書 2023ⁱⁱでは、2020 年から 2022 年の情報セキュリティインシデントの種類別報道件数（図 1-1）をまとめており、こちらも増加傾向にあります。

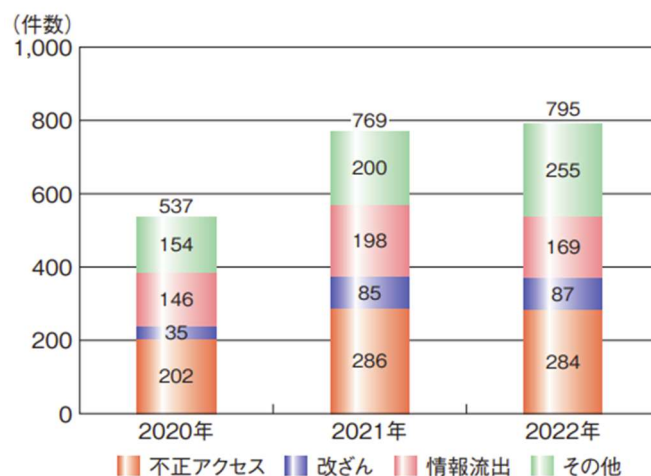


図 1-1: 情報セキュリティインシデントの種類別報道件数。

サイバー攻撃の影響は大きい

サイバー攻撃の増加は企業に対する脅威となります。具体的には、例えば 2023 年の名古屋港の事例が挙げられます。リモート接続機器の脆弱性を悪用した不正アクセスによるランサムウェア感染で、システムが約 2 日半にわたり停止し、業務に大きな影響を及ぼしました。

また、攻撃は多様化・巧妙化しています。IPA（独立行政法人情報処理推進機構）の情報セキュ

ⁱ <https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r05/html/nd24a210.html>

ⁱⁱ https://www.ipa.go.jp/publish/wp-security/t6hhco00000014r1-att/2023_All.pdf

リティ 10 大脅威 2024ⁱⁱⁱに挙げられた組織への脅威には、「サプライチェーンの弱点を悪用した攻撃」があります。サプライチェーンが狙われた事例では、2022 年のトヨタ自動車サプライチェーンへの攻撃が有名です。この攻撃によりトヨタ自動車では工場生産が一時停止しました。

サイバー攻撃が発生した際には、その対応には時間とともにコストがかかります。例えば IBM Security によるデータ侵害時にかかるコストに関する調査 (2023 年) ^{iv} vでは、データ侵害時の時間およびコストが言及されています。まず時間に関しては、組織がデータ侵害の発生を発見するまでの所要時間が平均 204 日、データ侵害の特定後にそれを解決するまでの所要時間が平均 73 日とのことです。さらに、データ侵害時の総コストは、グローバル平均で 445 万米ドル、日本平均では 452 万米ドルです。

サイバー攻撃は罰金に繋がる場合もある

EU では、EU 域内の個人情報保護を目的に、2018 年から一般データ保護規則 (General Data Protection Regulation、GDPR) ^{vi}が施行されています。この規則の下では、個人のデータは明確に特定された正当な目的のためにのみ収集・処理され、データに対する適切な安全性確保が必要です。また、データ侵害を認識したら、原則として 72 時間以内に、監督機関に報告する義務が課せられます。これに違反した場合、罰金として、最大 2000 万ユーロまたは前年度の世界年売上高の 4%のいずれか高い方が科せられます。

また、2025 年から施行予定の EU サイバーレジリエンス法^{vii}があります。これは、デジタル要素を備えた製品の製造業者および開発業者に対して課され、製品が市場に出る前に一定のセキュリティ基準を満たすことを目指しています。この法律の下では、インシデント発生時に、初期報告を 24 時間以内、詳細な報告提出をその後 3 日以内に行う必要があります。違反した場合、罰金として、最大 1000 万ユーロまたは年間総売り上げの 2%が科せられます。

サイバー攻撃に備える必要がある

前述のとおり、サイバー攻撃は増加傾向の上、多様化しており、サイバーセキュリティの重要性はますます高まっています。また、企業に対する罰則が厳格化していく傾向にあり、サイバーセキュリティ対策が「当然あるべきもの」として認識され、「仕組み」として社会に実装されていることがわかつています。セキュリティ対策をしない企業は、市場からの信頼を失うだけでなく、ビジネス継続が難しくなる可能性があります。しかし逆に言えば、対策をすることで顧客に対する信頼性向上や競争力強化に繋がるとも言えます。セキュリティ対策を、長期的に見た投資と捉えることが大事です。

セキュリティが投資である点を掘り下げる前に、本書の対象読者はシステム開発者向け、つまり

ⁱⁱⁱ https://www.ipa.go.jp/security/10threats/nq6ept000000g22h-att/kaisetsu_2024.pdf。年間で社会的影響が大きかったと判断された脅威を、個人・組織それぞれに対し 10 個紹介する。

^{iv} <https://www.ibm.com/reports/data-breach>

^v 16 か国で発生した 553 件を超えるデータ侵害インシデントのデータをもとに算出されている。

^{vi} <https://www.ppc.go.jp/enforcement/infoprovision/EU/>

^{vii} <https://digital-strategy.ec.europa.eu/en/policies/cyber-resilience-act>

ベンダ側の人物ですので、システムベンダにおけるセキュリティに関して確認しましょう。

1.1.2 システムを作る側の責任

システムを作る側のセキュリティ

自社をサイバー攻撃から守るセキュリティはもちろん重要ですが、自社がシステムベンダなら、顧客に提供するシステムのセキュリティも重要です。顧客に提供するシステムが保有する脆弱性は、攻撃者に利用され、顧客の事業被害（ビジネス中断や情報漏洩など）に繋がるリスクをもたらします。ゆえに、システムベンダは顧客に提供するシステムに責任を負うことになります。

ベンダとしては、「顧客は『セキュリティは、効果が理解しにくく、なくてもすぐには問題にならないので、そこに投資をしたくない』と言う」「顧客要求外のものに、お金も人も時間もかけられない」という状況に悩んでいるかもしれません。しかし、開発するシステムのセキュリティ対策を「顧客から言われていないから対応しない」と考えると、結果としてベンダ側にとっても重大なリスクに繋がります。

システムを作る側の責任

システムベンダ側のセキュリティ対策責任が問われたある裁判事例（2011年）^{viii}を紹介します。インテリア用品販売会社（原告）は、商品受注サイトから顧客のクレジットカード情報流出の被害があったことを受け、同サイト発注先のシステムベンダ（被告）に対し損害賠償を求めました。裁判所は、システムベンダ（被告）が必要なセキュリティ対策を講じる義務（債務）を負っていたことを認め、それを怠ったために債務不履行が成立すると判断し、被告のシステムベンダに対して損害賠償金の支払いを命じました。

上記判決は、システムベンダが顧客に提供するシステムに対して適切なセキュリティ対策を講じる責任を持つことを端的に表しています。セキュリティ対策は「オプション」ではなく「必須の要件」と言えます。システムベンダは、責任をもってセキュアなシステムを提供するようにしましょう。

システムを作る側がセキュリティ対策を行う利点

責任ばかりを語るのではなく、前項でも述べたようにセキュリティ対策を行う利点にも焦点を当てましょう。システム設計・開発を行う企業がシステムにセキュリティ対策を行う利点は为什么呢。

まず、セキュリティ対策が適切に施されたシステムは、顧客が安心して利用でき、顧客との長期的な関係構築に寄与するため、顧客や関連先からの信頼を獲得しやすくなります。またセキュリティ対策が適切に施されたシステムは、他社のセキュリティ対策が適切に施されていないシステムに対しての差別化となり、市場での競争力が高まります。高いセキュリティ品質を維持すると、自社のブランドの信頼性と評判が向上します。これにより新規顧客の獲得や、既存顧客のロイヤルティ

^{viii} https://www.softic.or.jp/semi/2014/5_141113/op.pdf。

強化、すなわち自社製品の継続的な利用につながります。

次に、セキュリティ対策が適切に施されたシステムは、各種の法規制や業界標準に準拠することが容易になります。特に GDPR や CCPA（California Consumer Privacy Act: カリフォルニア州消費者プライバシー法）などのデータ保護規制に対応するためには、サイバーセキュリティ対策が不可欠です。

これらの利点により、システム設計・開発を行う企業がサイバーセキュリティ対策を強化することは、製品の採用可能性を高めるだけでなく、全体的なビジネスの成功に寄与する重要な要素となります。

1.1.3 開発初期段階でのセキュリティ検討

システムを作る側（開発現場）の状況

システムベンダにはセキュアなシステムを開発・提供する責任や関連する利点があるものの、コスト・納期・機能性を優先しセキュリティまで手が回らない開発現場が多いというのもまた事実です。例えば、目下の納期を守るため、すぐには問題にならないセキュリティ対策を後回しにして結局対応しなかった、バグの修正や新機能の実装といった日々の業務に追われる中で、セキュリティアップデートの確認や最新のセキュリティ脅威への対応策の学習には時間を取れない、といった具合です。

開発初期段階でのセキュリティ検討

システムベンダ側のシステム開発チームが「多忙なので、開発時にはセキュリティを考慮しない」となってしまうことはぜひとも避けたいです。そこで、本書では「開発初期」からセキュリティを考慮する」という提案をします。やらなくてはいけないセキュリティ対策を「時間がないから」と後回しにすると、手戻りのために想定より多くの時間がかかります。反対に、最も手戻りが少ない開発初期にセキュリティを考慮すれば、セキュリティ対策にかかる時間は短く済むと考えます。

以下で詳細を説明します。まず、セキュリティ欠陥（脆弱性）に限らず、ソフトウェアの欠陥を修正するコストは、後工程になる程大きくなることが知られています^{ix} x。ソフトウェア開発における欠陥修正の相対コストに言及した文献^{xi}では、テスト段階で発見された欠陥の修正コストは、設計段階の場合の 15 倍とされています。リリース後の修正コストはそれ以上となります。

修正コストが大きくなるのは、下記のような多岐にわたる工程が追加発生するためです。

- ✓ 脆弱性の再現性や発生条件の特定

^{ix} <https://www.lawfaremedia.org/article/bolt-vs-baked-cybersecurity>

^x

[https://www.researchgate.net/publication/255965523 Integrating Software Assurance into the Software Development Life Cycle SDLC](https://www.researchgate.net/publication/255965523_Integrating_Software_Assurance_into_the_Software_Development_Life_Cycle_SDLC)

^{xi} 脚注 ^x に同じ。

- ✓ 修正すべきかの判断（修正しない場合の影響度、修正した場合の影響度をそれぞれ考慮）
- ✓ 実際の修正（コードの再設計、実装およびドキュメント）
- ✓ 修正後のテスト（脆弱性発見した段階までのテストを再度実施、リグレッションテスト^{xii}）

システム設計におけるセキュリティ欠陥の修正に関しても、これと同様と考えられます。追加コストを低減するためには、「脆弱性の検出を実装中/後のなるべく早期段階で行うこと」、「そもそも設計段階で脆弱性を埋め込みにくい設計をすること」が重要になります。

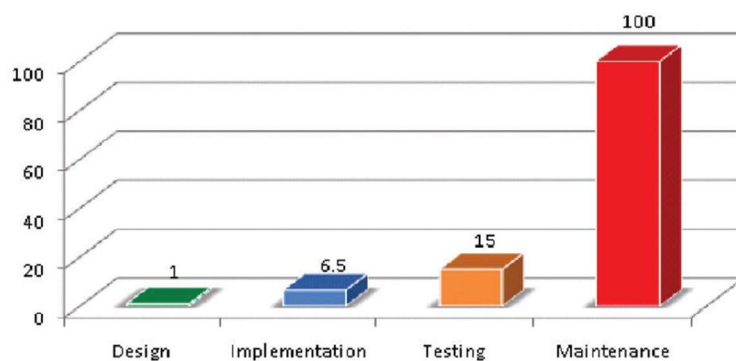


図 1-2: ソフトウェアの欠陥修正の工程別相対コスト。”INTEGRATING SOFTWARE ASSURANCE INTO THE SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)”より。

^{xii} システムの再リリースの際に既存機能が正常に機能することを確認するためのテスト。

セキュリティ・バイ・デザイン

セキュリティ・バイ・デザインとは、システムやソフトウェアの要件定義・設計の段階からセキュリティを組み込むアプローチです。この考え方では、セキュリティを後付けするのではなく、最初からシステムの一部と考え、設計・実装します。つまり、システム開発のあらゆる工程でセキュリティを確保することを目指します。これにより、脆弱性を早期に発見・修正することができ、システム全体のセキュリティレベルを向上させることができます。

基本的な考え方は以上の通りですが、「セキュリティ・バイ・デザイン」は考え方を指す呼称であるため、定義する組織・人物、場面に応じて若干ニュアンスが異なります^{xiii}。様々な組織がセキュリティ・バイ・デザインの定義を紹介しています。

- ✓ デジタル庁

(政府情報システムの) 企画工程から設計工程、開発工程、運用工程までを含めたすべてのシステム開発ライフサイクルにおいて、一貫したセキュリティを確保する方策^{xiv}

- ✓ 内閣サイバーセキュリティセンター (NISC)

情報セキュリティをシステムの企画・設計段階から確保するための方策^{xv}

- ✓ 米国サイバーセキュリティ・インフラセキュリティ庁 (CISA)

Secure By Design

悪意あるサイバー攻撃者による、デバイス、データ、接続されたインフラへのアクセスを合理的に保護した上で技術製品が構築されていること^{xvi}

本書で採用する定義

本書におけるセキュリティ・バイ・デザインの定義では、内閣サイバーセキュリティセンターの「情報セキュリティをシステムの企画・設計段階から確保するための方策」を採用しました。

^{xiii} 各組織で、どのようにセキュリティを設計プロセスに組み込むべきかが異なると言える。

^{xiv} [https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/e2a06143-ed29-4f1d-9c31-](https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/e2a06143-ed29-4f1d-9c31-0f06fca67afc/7e3e30b9/20240131_resources_standard_guidelines_guidelines_01.pdf)

[0f06fca67afc/7e3e30b9/20240131_resources_standard_guidelines_guidelines_01.pdf](https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/e2a06143-ed29-4f1d-9c31-0f06fca67afc/7e3e30b9/20240131_resources_standard_guidelines_guidelines_01.pdf)

^{xv} https://www.nisc.go.jp/pdf/policy/general/SBD_overview.pdf

^{xvi} <https://www.cisa.gov/securebydesign>

開発初期とはいつなのか

セキュリティ・バイ・デザインでは開発初期段階に脆弱性を埋め込まないことを目指しますが、開発初期とは具体的にいつを指すのかを定義しておきましょう。そのために、ここでは開発モデルとしてV字開発モデルを考えます。

システム開発モデルの1つであるV字開発モデルは、ウォーターフォールモデル（1つの開発工程を完了させてから次の工程に進むモデル）の拡張^{xvii}と考えられます。開発プロセスと同時に、それに対応するテスト計画を進めることを重視します。図1-3はV字開発モデルの概念図です。表1-1がV字開発モデルの各段階における実施内容を説明しています。各工程の最後には、次の工程に進んでよいかを、成果物から判断します。成果物は、例えば、設計であれば設計書、コーディングであればコードです。それゆえ、この開発モデルでは基本的に手戻りが発生しないことになっています。

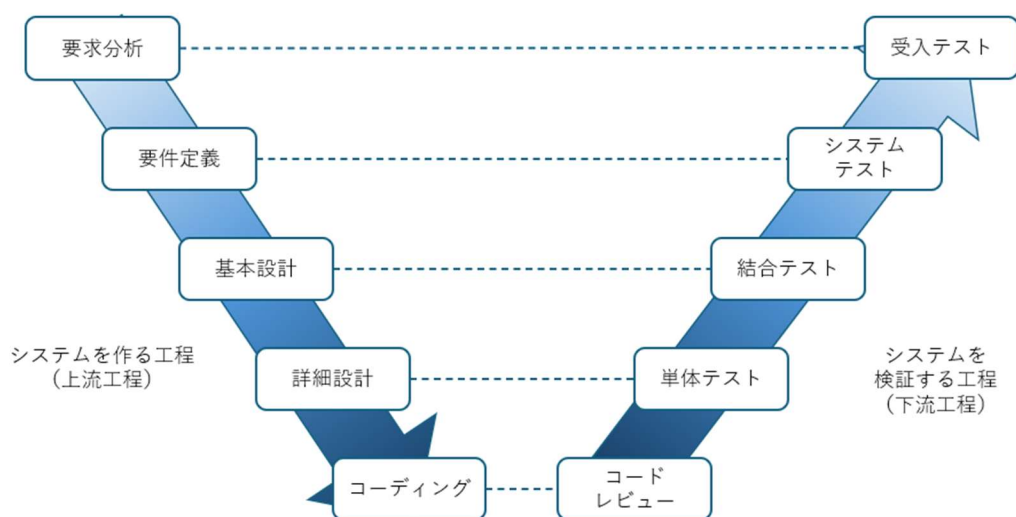


図 1-3: 開発のV字モデル。左矢印：開発プロセス、右矢印：テストプロセス。各四角形が各工程を表す。点線は、開発プロセスの各工程実施時に、対応するテスト工程の計画を立てることを表す。

^{xvii} ウォータフォール開発と比較してアジャイル開発では、短く区切った期間（イテレーション）を繰り返します。つまり、その短い期間の中でV字モデルの手順に従うイメージです。つまり、ウォーターフォール開発ほど直接的でないにせよ、アジャイル開発にもV字モデルは適用可能です。

表 1-1: 開発の V 字モデルの各段階詳細。「開発者の関与度」は、筆者らの今までの経験から、開発チームが関与できるかどうかの指標を示す。○:「関与できる」、△:「関与できない場合がある」。

段階	内容	開発者の関与度
要求分析	システムが解決すべき問題や満たすべき顧客やユーザのニーズを特定し、理解するプロセス。 ・ ニーズの情報収集 ・ 要求を整理し、機能・非機能・制約条件に分類 ・ 要求の一貫性、実現可能性を検証する	△
要件定義	要求分析で明らかになった要求をもとに、システムの仕様や機能（要件）を具体的に定義するプロセス。 ・ 仕様書の作成 ・ 合意の形成、変更管理	○
基本設計	要求定義に基づき、システム全体の構造や機能を明確にするプロセス。 ・ システムアーキテクチャ設計 ・ ユーザインタフェース設計 ・ データ設計 ・ セキュリティ設計 等	○
詳細設計	基本設計で定義された仕様をもとに、各コンポーネントの内部構造や具体的な実装方法を設計するプロセス。 ・ アルゴリズム設計 ・ DB やその他ストレージの構造を詳細化 等	○
コーディング	詳細設計で定義された仕様や構造に基づいて、実際にプログラムを書くプロセス。	○
コードレビュー	他の開発者が書いたプログラムがコーディング規約（変数名、コメント等）に従っているかを確認するプロセス。	○
単体テスト	ユニット（最小単位）である機能やメソッドが意図した通りに動作することを確認するプロセス。	○
結合テスト	単体テストで検証された個々の機能を組み合わせ、それらが正しく連携して動作することを確認するプロセス。	○
システムテスト	統合されたシステム全体が仕様通りに動作するかを確認するプロセス。	○
受入テスト	システムが顧客の要求を満たしているか確認するプロセス。	△

ちなみにアジャイル開発の場合でも、大幅な変更が必要になる可能性があります。その場合でも、一度に全体をやり直すことはありませんが、新たな要件や変更点をバックログに追加し、影響や優先順位を再評価し、イテレーション計画の調整が必要になります。

具体的なプロセス

セキュリティ・バイ・デザインでは、リスク評価、セキュアな設計、セキュリティテスト、継続的なモニタリングなどのプロセスを通じて、システムのあらゆる段階でセキュリティを確保することを目指します。

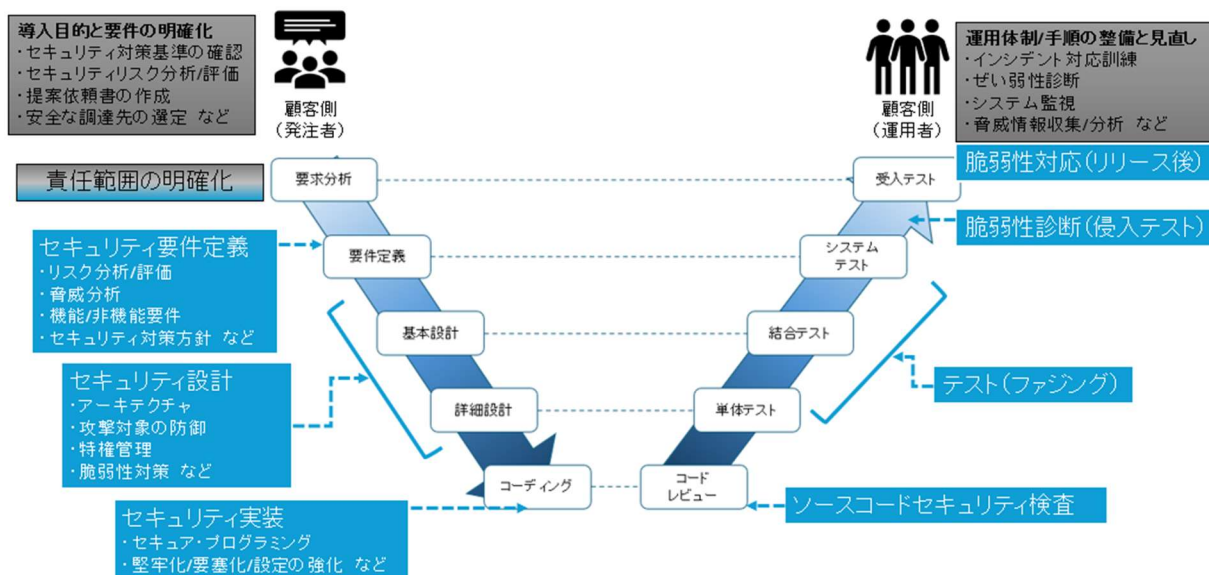


図 1-4: V字開発モデルの各段階におけるセキュリティ対策の例。

次に、V字開発モデルの各段階で実施可能なセキュリティ対策を図示しました（図 1-4）^{xviii}。

図中の黒ボックス部分は一般的に顧客側で実施するセキュリティへの対応を示しています。また、青ボックス部分は開発チーム側が実施すると考えられるセキュリティへの対応です。

- セキュリティ要件定義
 - システムのセキュリティリスクを軽減しシステムを保護することを目的に、具体的なセキュリティ要件を明確化します。
- セキュリティ設計
 - 脅威や攻撃に対して効果的に防御することを目的にセキュリティのアーキテクチャやデータ保護の方法を決定します。
- セキュリティ実装
 - 開発するコードがセキュリティ脆弱性を含まず、確実に動作することを目的に、コーディングガイドラインに従った記述をします。

^{xviii} [https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/bba70a3b-c698-46ba-9b5f-](https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/bba70a3b-c698-46ba-9b5f-078e1623a181/c34f77e8/20240329_policies_development_management_outline_02.pdf)

[078e1623a181/c34f77e8/20240329_policies_development_management_outline_02.pdf](https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/bba70a3b-c698-46ba-9b5f-078e1623a181/c34f77e8/20240329_policies_development_management_outline_02.pdf)。デジタル庁のセキュリティ・バイ・デザインガイドラインを参考に、筆者らが作成。図中に記載されている内容は網羅性を保証するものではありません。

- コードレビュー
 - コードの品質向上を目的に、バグやセキュリティ脆弱性を発見し、修正します。
- テスト段階（単体テストからシステムテスト）
 - システムが設計通り動作することの確認を目的に潜在的なセキュリティ脆弱性を発見・修正します。
- 脆弱性診断（侵入テスト）
 - システムのセキュリティ脆弱性を攻撃者の視点から発見し、修正することを目的に、攻撃シナリオのシミュレーションを行い、セキュリティリスクの評価を行います（実施を義務付けられている業界も存在します）。
- 脆弱性対応（リリース後）
 - 顧客との契約によっては、システムを納品した後も保守業務に関わる場合があります。リリース後（開発工程における保守段階）に報告される脆弱性については、製品のセキュリティを担当する **PSIRT（Product Security Incident Response Team）** またはそれに相当する部署が対応します。これらの部署は、脆弱性の発見、トリアージ、修復、開示などのプロセスを実施し、システムの安全性を維持することを目的としています。今後、**SBOM（Software Bill of Materials）** の導入を通じて、脆弱性管理の強化を検討している組織もあるかもしれません。

開発初期段階でのセキュリティ関連コスト見積

システム開発初期でのセキュリティ考慮に関連して、システム開発初期におけるセキュリティ関連のコスト見積に触れておきます。企業によっては、そもそも開発チームがコスト見積に関与できないかもしれませんが、可能なら、見積段階でセキュリティに必要コストを計上しておくことが重要です。なぜなら、後からセキュリティ機能が必要であるとわかって、それを実現するためのリソース（開発およびテストに必要な費用および時間）がなく、仕様変更に伴う追加発注や納期の変更が必要となる事態になりかねません。

1.2 本書について

ここまでは、セキュリティの重要性に加え、セキュリティ・バイ・デザインの考え方をを用いて、コスト負担を低減しながらシステムにセキュリティ対策を組み込む提案について説明してきました。本書はセキュリティ・バイ・デザインを開発初学者に伝えるための入門書です。ここからは、本書の作成目的や対象読者等について記載します。

1.2.1 本書の作成目的・対象読者

本書は、開発初学者に、初学者の頃から「セキュリティ・バイ・デザインの意識」を養ってもらうためのものです。開発初学者は配属 1～3 年目程度とし、配属 4 年目までには必ず知っておいてほしい内容を記載しました。対象読者の詳細は下記にまとめます。

- 対象読者：
 - システム開発チームに配属された開発初学者（1～3 年目）。
 - ✧ 設計・開発者
 - ✧ 品質保証担当者
- 読了後に期待する効果：
 - セキュリティ初学者の開発者でも、セキュリティ意識を持って開発に取り組む。
 - 発注側や上流工程のセキュリティ要件／対策が不十分でも、受入れ時に「なんか変だ」と疑問をもち、確認や修正提案ができる。

1.2.2 本書の特徴・スコープ・使い方

誰でも気軽に読める入門書に限ると、一般向けの書籍がとて多く、開発者向けの書籍は少ないという意見が本プロジェクト内で出ました。そこで、本書は初学者にも短く読みやすい読み物を目指しました。

アンチパターン

読みやすくするために、アンチパターンを使いました。アンチパターンとは、失敗事例から教訓を得る形式です。開発チームのありがちなミスがセキュリティ事故に繋がるというストーリーから、初学者でもすんなりとセキュリティを学習できると期待しています。

アンチパターンの題材選定

本書のアンチパターンの題材は下記観点で選定しました。

- ✓ システム開発業務において、新人（業務経験 3 年目まで）に特に注意してほしい、または実際に直面しそう。
- ✓ 理解がそれほど難しくない。

また、V 字開発モデルの各工程で個数が偏らないよう調整しました。

一方で、題材をセキュリティ上の欠陥の種類を網羅するように選定することもできました。セキ

セキュリティ専門家が作成する OWASP Top10^{xix}や CWE Top25^{xx}などのリスクランキングは、セキュリティ欠陥の発見頻度や危険度に基づいた選定も有効ですが、初学者が読む上では本書の選定方法の方が向いていると考えました。しかし、学習を続ける上で、有名な脆弱性の知識を網羅的に学習することは避けられないため、上記ランキングは、この本を読み終えた後にぜひ読んでみてください。

アンチパターンの個数

また、アンチパターンの個数は 12 個としました。その理由は、月に 1 個、年に 12 個のペースで利用可能にするためです。セキュリティには学習すべきことが非常にたくさんあります。いきなりすべてを学習するようと山ほどの文献を渡されても、「セキュリティ学習のはじめの一步」の前に挫折してしまう可能性があります。まずは忙しい業務の中でも無理なく、月に 1 個のペースでセキュリティ知識に触れてほしいと考えました。

注意

本書は初学者向けということもあり、また紙幅の関係からも、セキュリティ対策を網羅的に記載するものではありません。すなわち「本書の内容を理解・実施すれば、セキュリティは完璧！」という訳ではありません。

本書は、個別の事例ベースで具体的である分、セキュリティ対策を網羅的に学習することには向いていません。セキュリティ対策を網羅的に学習するためには、一般公開されたガイドラインや、皆さんの所属する組織ごとのポリシー・戦略などのドキュメントを読む必要があります。本書はその事前学習として利用されることを期待しています。

^{xix} <https://owasp.org/Top10/ja/>。Open Worldwide Application Security Project の略。Web アプリケーション・セキュリティに関するリスクについてのランキングを提供している。

^{xx} <https://cwe.mitre.org/top25/index.html>。Common Weakness Enumeration（共通脆弱性タイプ一覧）。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を提供している。

第2章 アンチパターン

2.1 アンチパターンの見方

本書では、開発に関連する計 12 個のアンチパターン（表 2-1）を紹介します。どのアンチパターンも、開発チームのありがちなミスがセキュリティ事故に繋がることを伝えています。図 2-1 のように、アンチパターンは V 字開発モデルの要件定義から詳細設計という順番で並べていますが、各アンチパターンは独立しており、気になる項目から読み始めることもできます。

表 2-1: 12 個のアンチパターン。

No.	タイトル	キーワード
1	セキュリティと日常のアナロジ	リスクアセスメント、リスク評価
2	曖昧な言葉で書かれたドキュメント	設計書、ドキュメントライティング
3	専門家不在のセキュリティレビュー	レビュー
4	権限設定の見落とし	AAA、アクセス制御
5	思いがけない内部犯	AAA、サプライチェーンセキュリティ
6	ホワイトリストの登録漏れ	不正プログラムの動作禁止、関係者の連携
7	消えたログ	ランサムウェア被害、ログシステム設計
8	システムバックアップの落とし穴	脆弱性管理、データ保護
9	曖昧なセキュリティ設定がもたらすリスク	セキュリティ・バイ・デフォルト、責任分界点
10	不要サービスはどこにある？	デバッグツールとテストツールの停止
11	品質寄りのリリース判定	品質管理、脆弱性
12	自作コードへの過信	セキュアコーディング、フレームワークの利用

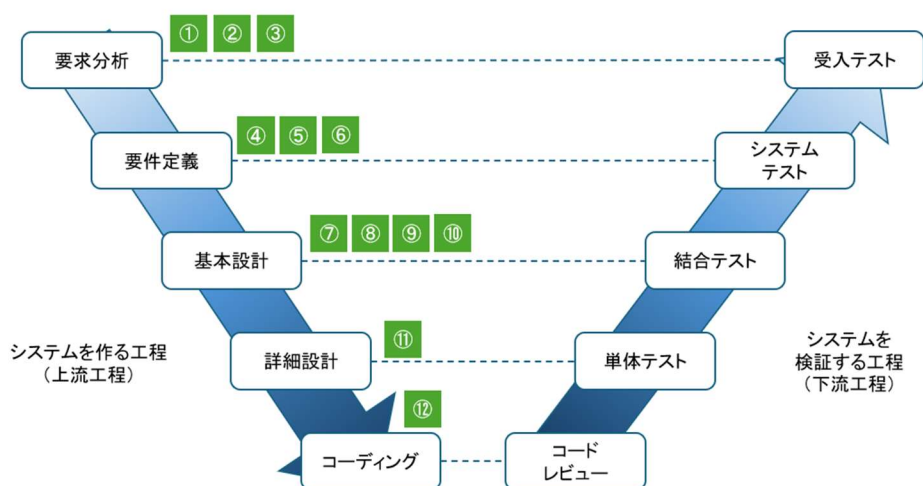


図 2-1: アンチパターンと V 字開発モデルの対応。

アンチパターンの読み方

①タイトル

1) セキュリティと日常のアナロジ

このページでは、これからセキュリティを学ぶ人のために、「セキュリティが考慮されていない設計」と「セキュリティが十分に考慮されている設計」について例え話を挙げて説明します。セキュリティは考慮すべき項目が多岐にわたる。様々な分野の知識が必要であるため、全てを網羅を持つ人はいらっしゃると思います。たしかにセキュリティは奥が深いですが、基本的な考え方は非常にシンプルです。是非このアンチパターンから、「セキュアな設計」とは何かについて考えてみてください。

②事例

事例

あなたは顧客に依頼されて「駐車場に設置するゲート」を設計することになりました。顧客から必要機能や予算、納期などを聞き、すべてを満足する「駐車場に設置するゲート」を設計しました。ある日、あなたは顧客のもとを訪れ、自身が開発した「駐車場に設置するゲート」を説明しました。それが以下の図通りであったとします。




図1: 駐車場のゲートで考えられるアンチパターン

さて、この図の中にある問題を列挙してください。なぜそれが問題だと思ったのかについて、「どのような状況で、誰・何が、どのような動作をしたときにどのような影響が現れるか」をメモしてください。

③ヒント

ヒント: 決定する駐車場のゲートは全てで済みます。そして、ある人が駐車場の管理人である場合、どのようなことを想定する必要がありますか？

④解説

解説

まずゲート脇の道を走行して、ゲートを介さず駐車場へ侵入する。なぜ発生するか？

- 歩道の幅、急いでいるから。
- 駐車料金を支払いたくないから。

問題点

- 駐車場を利用する際に、ゲート付きの道の通過を強制できていない。
- たまたま道路脇の芝生より歩道があるため、タイヤ側から道路脇に突っ込みやすいが、そうであれば道路脇に突っ込むにいい。
- 道路脇に突っ込んだとしても、顧客には犯人がわからず、捕らえられない。
- この駐車場ゲートには顧客要求を満足する機能が搭載されていることですが、根本的な構造上の問題を抱えていることが分かります。
- この問題を「セキュリティ上の問題があるのか？」と問った方もいらっしゃると思いますが、そのような状況のようには考えてください。「運転手がゲートを認識しても、顧客は道路脇に突っ込まない」→「管理者が認識を回避しても、管理者は道路脇に突っ込まない」といってしまえば、いかにゲートが壊れてもゲートを設置した駐車場がセキュアとは限りません。駐車場利用者が、あなたの期待する行動（ゲートからの出入り）を取らない限りはなりません。このように、駐車場全体を制御したうえで設計する必要があります。
- 対策に、システム側へ常に監視的なセキュリティ機能（犯罪を察知して、他の部分に逃げ遅れがあれば、急遽としてセキュリティが機能になります。攻撃者は「セキュリティの最も弱い場所（Weak Link）を握る」ためです。サイバーセキュリティを考える際、広い視点をもって、特に脆弱な部分に目を向けてください。

対策

「駐車場ゲート」への対策を考えましょう。駐車場に入る／から出るすべての車にゲート通過を強制するにはどうすればいいでしょうか。対策の一例として、車が芝生に侵入できないよう柵を設置する対策が考えられます。また、監視者を確保するための対策では、監視カメラ設置し、あとでその映像を分析する手法もあります。

万能的なセキュリティ対策は存在しないため、それぞれの状況に応じて最適な対策を講ずる必要があります。

⑤教訓

⑥対策

① タイトル

アンチパターンのタイトルです。

② 事例

システム開発の現場で、検討漏れや判断ミスによってセキュリティ事故が発生するストーリーを紹介します。読者が具体的にイメージしやすいように工夫しています。

③ ヒント

読者にセキュリティ事故の原因について考えてもらうため、問題提起をしています。

④ 解説

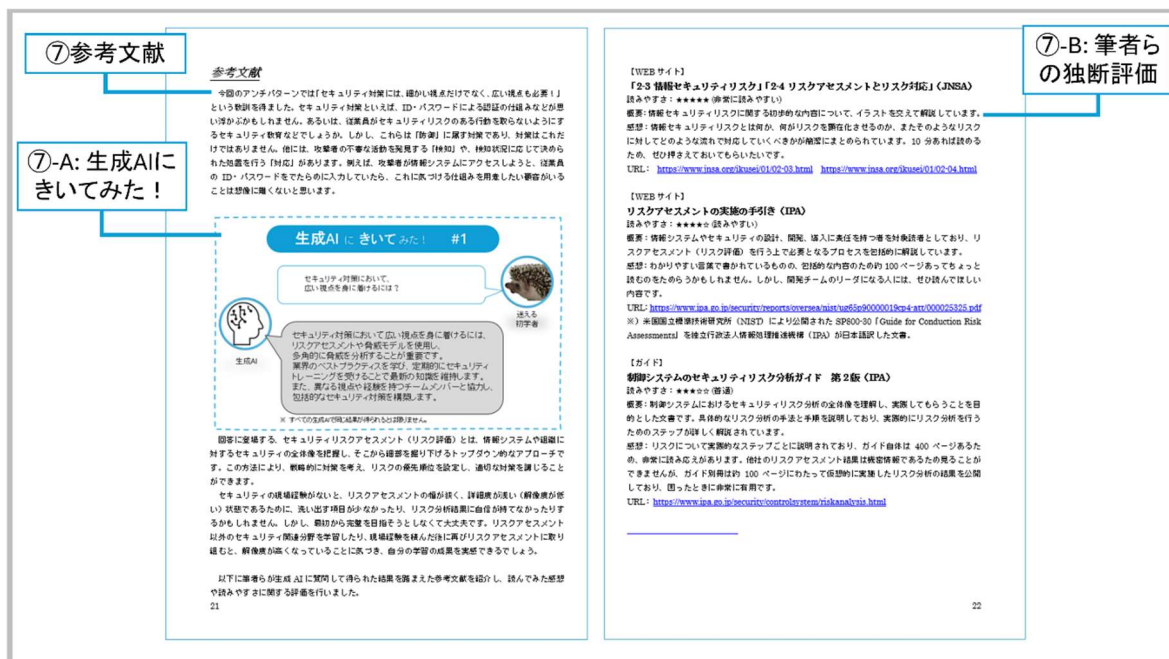
セキュリティ事故が発生した原因について解説しています。

⑤ 教訓

セキュリティ事故とその原因をもとに、今後の業務に役立ててほしい教訓を提示しています。

⑥ 対策

事故に対する一般的な対策を簡単に紹介し、詳しい解説や対策は参考文献に委ねています。



⑦ 参考文献

このアンチパターンに興味を持ち、詳細をもっと知りたいと思った読者のために「次に読むべき参考文献」を紹介しています。

A: 生成 AI にきいてみた！

読者が「教訓」から疑問に思うであろうことを、筆者らの相棒である生成 AI に質問したので、その回答を記載します。生成 AI は疑問に対する一般的な回答を返すため、それを膨らまして、参考文献を考えました。

注) すべての生成 AI で同じ結果が得られるとは限りません。

B: 筆者らの独断評価

参考文献に対する独断評価で、読みやすさ、概要、感想（筆者が読んだ際の）を記載しています。読みやすさは、参考文献を読む上で前提となる専門知識の量、文章自体の難易度、ページ数などの観点から、筆者らの独断と偏見により星 5 段階で評価しています。詳細は下記を参照ください。

読みやすさの目安

- ★5：非常に読みやすい（極めて簡潔で理解しやすい。専門知識がなくても読める）
- ★4：読みやすい（基本的にはわかりやすく、主張の大多数が理解できる）
- ★3：普通（専門用語が一部あるものの、基本的な知識があれば問題なく理解できる）
- ★2：やや読みづらい（専門用語や難解な表現が多く、特定の知識がないと理解が難しい）
- ★1：非常に読みづらい（専門家や高い知識を持つ人向け）

注) 文献の構成が独立していて読みやすい場合、ページ数は多いが図解が多く読みやすい場合などは、評価が 1 段階上がることがあります。

2.2 アンチパターン集

次ページからアンチパターンを紹介していきます。

1) セキュリティと日常のアナロジ

このページでは、これからセキュリティを学ぶ人のために、「セキュリティに穴がある設計」について、例え話（アナロジ）を使って説明します。セキュリティは考慮すべき項目が多岐にわたり、様々な分野の知識が必要であることから苦手意識を持つ人がいるかもしれません。確かにセキュリティは奥が深いですが、基本的な考え方は意外とシンプルです。ぜひこのアンチパターンから、「セキュアな設計」とは何かについて考えてください。

事例

早速ですが、あなたは顧客に「駐車場の入出場ゲート」の設計を依頼されました。必要な機能・予算・納期などを聞いたあなたは、すべてを満足するゲートを設計しました。ゲートは設計通りに完成し、無事駐車場に設置されました。

暫く経ったある日、あなたはその駐車場を訪れる機会がありました。そこで自身が設計したゲート（図 2-2）を見つけましたが、その様子に非常に驚きました。



図 2-2: 駐車場ゲートⁱ。

ここで、図から、このゲートにどんな問題事象が起こっているか推測してみましょう。また、「なぜ」そのような事象を発生するのでしょうか。

ヒント：芝生の上の雪にはタイヤ痕があります。これは何を表すのでしょうか？

ⁱ

<https://wiki.sei.cmu.edu/confluence/display/seccode/Top+10+Secure+Coding+Practices?focusedCommentId=88044413>

解説

事象：駐車場を利用する車が、ゲートを通過せずに、ゲート脇の芝生を通過して、出入りする。

なぜ発生するか？：

- ✓ 駐車料金を支払いたくないから。
- ✓ 平日の朝、急いでいるから（駐車料金を払う意思はあるが、ゲート通過の時間が煩わしい）など

問題点：

- ✓ 駐車場を利用する車に、ゲート通過を強制できていない。本来、ゲートを迂回されないような対策が必要。
- ✓ たまたま迂回路が芝生かつ積雪があるため、タイヤ痕から迂回行為に気づきやすいが、そうでなければ駐車場管理者は迂回行為に気づくことが難しい。もし迂回行為に気づけたとしても、誰が犯人かわからず、罪に問う事ができない。

いかがでしょうか？駐車場ゲートは顧客要求を満足したかもしれませんが、駐車場全体としては欠陥を抱えていることがわかります。強固なゲートを設置した駐車場がセキュアとは限りません。駐車場利用者が期待する行動（ゲートからの出入り）を取らないかもしれません。ゲート機能のみに注目するのではなく、駐車場全体を俯瞰で見て、利用者の行動も考えて、設計を実施する必要があります。

上記の駐車場での事例をもとに、サイバーセキュリティ対策で発生する問題に読み替えてみましょう。

事象：悪意ある者が、認証の機構を迂回して、社内システムに侵入する。

なぜ発生するか？：悪意ある者が社内の機密情報を盗み出したいから。 など

問題点：

- ✓ 社内システムにアクセスするネットワーク経路が一元化されていない。
- ✓ 攻撃者が認証を回避しても、監視機構がないため管理者は認証回避に気が付かない。

システムの一部に強固なセキュリティ機能を導入していても、他に抜け漏れがあれば、全体としてセキュリティが脆弱になります。攻撃者は「セキュリティの最も弱い場所（Weakest Link）を狙う」ためです。サイバーセキュリティでも、広い視点を持ち、脆弱な部分に目を向けてください。

教訓：セキュリティ対策には、細かい視点だけでなく、広い視点も必要！

対策


「駐車場ゲート」への対策を考えましょう。駐車場への入出場するすべての車にゲート通過を強制するにはどうすればよいのでしょうか。対策の一例として、ゲートが設置された道の路肩にガードレール等を設置し、車を芝生エリアに侵入させないなどが考えられます。また、違反者を確認するための対策では、監視カメラ設置して車のナンバーを記録できれば、その情報を頼りに車の所有者を特定する方法もあります。

ちなみに、万能なセキュリティ対策というものは存在しないため、状況に応じて最適な対策を導き出す必要があります。


参考文献

今回のアンチパターンでは「セキュリティ対策には、細かい視点だけでなく、広い視点も必要！」という教訓を得ました。セキュリティ対策といえば、ID・パスワードによる認証の仕組みなどが思い浮かぶかもしれませんが。あるいは、従業員がセキュリティリスクのある行動を取らないようにするセキュリティ教育などでしょうか。しかし、これらは「防御」に属す対策であり、対策はこれだけではありません。他には、攻撃者の不審な活動を発見する「検知」や、検知状況に応じて決められた対応を行う「対応」があります。例えば、攻撃者が情報システムにアクセスしようと、従業員の ID・パスワードをでたらめに入力していたら、これに気づける仕組みを用意したい顧客がいることは想像に難くないと思います。

生成AI に きいて みた！ #1



生成AI



迷える
初学者

セキュリティ対策において、
広い視点を身に着けるには？

セキュリティ対策において広い視点を身に着けるには、リスクアセスメントや脅威モデルを使用し、多角的に脅威を分析することが重要です。業界のベストプラクティスを学び、定期的にセキュリティトレーニングを受けることで最新の知識を維持します。また、異なる視点や経験を持つチームメンバーと協力し、包括的なセキュリティ対策を構築します。

※ すべての生成AIで同じ結果が得られるとは限りません。

回答に登場する、セキュリティリスクアセスメント（リスク評価）とは、情報システムや組織に対するセキュリティの全体像を把握し、そこから細部に掘り下げていくトップダウン的なアプローチです。この方法により、戦略的に対策を考え、リスクの優先順位を設定し、適切な対策を講じることができます。

セキュリティの現場経験がないと、リスクアセスメントの幅が狭く、詳細度が浅い（解像度が低い）状態であるために、洗い出す項目が少なかったり、リスク分析結果に自信が持てなかったりするかもしれません。しかし、最初から完璧を目指そうとしなくて大丈夫です。リスクアセスメント以外のセキュリティ関連分野を学習したり、現場経験を積んだ後に再びリスクアセスメントに取り組むと、解像度が高くなっていることに気づき、自分の学習の成果を実感できるでしょう。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【WEB サイト】

「2-3 情報セキュリティリスク」「2-4 リスクアセスメントとリスク対応」(JNSA)

読みやすさ：★★★★★(非常に読みやすい)

概要：情報セキュリティリスクに関する初歩的な内容について、イラストを交えて解説しています。

感想：情報セキュリティリスクとは何か、何がリスクを顕在化させるのか、またそのようなリスクに対してどのような流れで対応していくべきかが簡潔にまとめられています。10 分あれば読めるため、ぜひ押さえておいてもらいたいです。

URL：<https://www.jnsa.org/ikusei/01/02-03.html> <https://www.jnsa.org/ikusei/01/02-04.html>

【WEB サイト】

リスクアセスメントの実施の手引き (IPA)

読みやすさ：★★★★☆(読みやすい)

概要：情報システムやセキュリティの設計、開発、導入に責任を持つ者を対象読者としており、リスクアセスメント（リスク評価）を行う上で必要となるプロセスを包括的に解説しています。

感想：わかりやすい言葉で書かれているものの、包括的な内容のため約 100 ページあってちょっと読むのをためらうかもしれません。しかし、開発チームのリーダになる人には、ぜひ読んでほしい内容です。

URL：<https://www.ipa.go.jp/security/reports/oversea/nist/ug65p90000019cp4-att/000025325.pdf>

※）米国国立標準技術研究所（NIST）により公開された SP800-30「Guide for Conduction Risk Assessments」を独立行政法人情報処理推進機構（IPA）が日本語訳した文書。

【ガイド】

制御システムのセキュリティリスク分析ガイド 第2版 (IPA)

読みやすさ：★★★★☆(普通)

概要：制御システムにおけるセキュリティリスク分析の全体像を理解し、実践してもらうことを目的とした文書です。具体的なリスク分析の手法と手順を説明しており、実践的にリスク分析を行うためのステップが詳しく解説されています。

感想：リスクについて実践的なステップごとに説明されており、ガイド自体は 400 ページあるため、非常に読み応えがあります。他社のリスクアセスメント結果は機密情報であるため見ることができませんが、ガイド別冊は約 100 ページにわたって仮想的に実施したリスク分析の結果を公開しており、困ったときに非常に有用です。

URL：<https://www.ipa.go.jp/security/controlsystem/riskanalysis.html>

2) 曖昧な言葉で書かれたドキュメント

このページでは、設計書などのドキュメントの書き方に関するアンチパターンを紹介します。設計書の作成目的は「プロジェクトに関する情報を整理・共有し、意思決定や判断の根拠を明確にすること」です。決定事項を文章として残しておくことは、すぐ後の工程でその設計書を読む人やテスターだけでなく、プロジェクト期間中およびその後の将来プロジェクトに関係する人たちにとっても大変意味のあることです。今回のアンチパターンから、もう一度「良いドキュメント」について考えてみてください。

事例

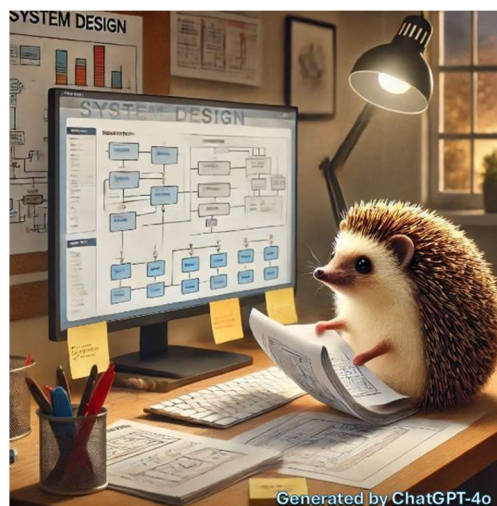
Aさんは設計書を作成することになりました。そこで要求仕様書を確認すると、セキュリティ要件として「安全に通信できること」と記載されていました。Aさんはセキュリティに明るくないながらも「通信を暗号化すること」が要求であると考え、以下のような設計書を作成しました。

※仕様書の一部抜粋

◆通信の暗号化

データの漏洩やデータの改竄を防止するために、通信の暗号化を行う。

- Web サーバと Web ブラウザ間の通信は HTTPS プロトコルを使用する。
- 営業所 A と営業所 B 間の通信は、IP-VPN（専用線）を使用する。



ヒント：顧客の要求仕様書に記載されていた「安全に通信できること」とは、本当に「通信を暗号化すること」で実現されますか？また、システムで使用するすべての通信を把握できていますか？

解説

要求仕様書に記載されていた「安全に通信できること」とは非常に曖昧な要求ですね。このアンチパターンでは、設計者の A さんが「通信を暗号化すること」であると解釈していました。もし、「X 社のネットワークスイッチ（型番 XXX-XXX）を使用すること」等の具体的で明確な要求が記載されていれば、設計者が解釈する余地はありません。そのネットワークスイッチを使用してシステムを構築し、意図した通りに動くことを確認すればよい訳です。

このアンチパターンには以下の 3 点の問題があります。

- ✓ 要求「安全に通信すること」には具体性がない。その文言が、機密性に関する「通信を暗号化すること」を指すのか、可用性に関する「通信経路を冗長化すること」を指すのか、もしくはその両方か、判断することができない。
- ✓ Web サーバ・WEB ブラウザ以外の記載がない。その他の通信が存在しないのか、それ以外は暗号化する必要がないのかが不明である。
- ✓ 「営業所 A と営業所 B 間の通信を IP-VPN（専用線）を使用する」ことを採用した。しかし、設計書内ではその方法に至った背景や根拠を説明していない。そのため、導入コストを抑えるためにインターネット VPN（一般的に IP-VPN（専用線）よりも安価）の採用を依頼されるかもしれない。

設計上の欠陥がなく、誤字脱字もない完璧なドキュメントを作成することはできません。設計者には様々な業務があり、業務に充てられる時間も有限です。しかし、特に重要な箇所、つまり明確にしておかないと後で大きな問題になる箇所は優先して対応すべきです。顧客の曖昧な要求は、後で大きな問題になる可能性が高いので、そのような箇所を特定し、修正工数を予想して優先順位付けを行い、密接なコミュニケーションを通じて合意を形成し文書化しましょう。

教訓：具体性がなく曖昧な文章は認識の齟齬を生む。

対策

システム開発には多くの関係者が関与するため、明確で分かりやすいドキュメントを作成することが重要です。これにより、情報共有とコミュニケーションがスムーズになります。曖昧な単語や表現を使用すると、誤解が生じて問題が発生する可能性があります。

対策として、一貫性のあるフォーマットを使用し、用語の定義を明確にし、簡潔で明解な表現を心がけることで、関係者が情報を迅速に理解できるようにします。


- ✓ 「安全に通信すること」という要求は、どこから出てきたのかを明確にしましょう。主語（誰と誰が通信するのか？）や状況（どのような条件下で？）、目的（ユーザ情報の保護のため）などを確認・明確にすることで、要求を正確に理解できるようになります。
- ✓ 設計ドキュメントでは、その方法を採用するに至った背景や、根拠を説明しましょう。例えば設計の各種代替案を網羅しそれらの強みと弱みを示すことで、設計ドキュメント作成者が（セキュリティを含む）仕様に対して注意を払い、理解した上で決定したと伝えることができます。

参考文献


今回のアンチパターンでは「具体性がなく曖昧な文章は認識の齟齬を生む」という教訓を得ました。ドキュメントを書くことが苦手だと感じているエンジニアの中には「技術的な内容を非技術者にもわかりやすく説明することが難しい」や「どのように書けばよいかの明確なガイドラインやフォーマットがなく、何をどのように書くべきかわからない」と感じている人がいるかもしれません。

一般的には、「分かりやすい文章を書く人のまねをする」や「対象読者をイメージする」ということで、「経験レベル (熟練プログラマ/初学者プログラマ)、ドメイン知識 (エキスパート/初学者)」を想定する方法があります。

生成AI に きいて みた！ #2



生成AI



迷える
初学者

優れた設計ドキュメントとは
どのようなものか教えて？

優れた設計ドキュメントは、明確で簡潔、かつ一貫性のある内容を持ち、技術的な詳細を分かりやすく説明します。
用語の定義を明確にし、設計の背景や決定理由を含み、将来の読者の知識レベルを考慮しています。
また、コードの一部として扱い、更新や保守が容易な形式で提供されるべきです。

※ すべての生成AIで同じ結果が得られるとは限りません。

開発プロジェクトには、経験レベルやドメイン知識が様々なメンバが参加しています。そのため、設計ドキュメント作成者が省略して書いた場合、意図した通りに伝わらない可能性があります。丁寧に説明しすぎると、設計ドキュメントの量が増え、レビューが大変になります。

一貫性のあるドキュメントは、わかっている人が読み飛ばせるような構造になっており、必要な部分だけを読めます。簡潔な文章はすぐに思いつくものではなく、短いドキュメントを書くためには、まず全情報を含む長いドキュメントを作成し、編集工程を通じて重複した情報を可能な限り取り除くことが多いです。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【WEB サイト】

LINE 社内で大評判のテクニカルライティング講座で説明した内容をあらためてブログにまとめてみた (LINE ヤフー株式会社)

読みやすさ: ★★★★★ (非常に読みやすい)

概要: この技術ブログは、わかりやすく文章を書くためのコツや方法の共有を目的とした文書です。文章のバリエーションの作成、主語と述語の明確化、文の短縮方法(「ので」「し」を避ける)、箇条書きや表の活用などが詳しく解説されています。

感想: 具体的な手順や事例を通じて、どのように改善するかがわかりやすく解説されており、短時間で読める構成になっています。

URL: <https://engineering.linecorp.com/ja/blog/line-technical-writing-course>

【書籍】

技術者のためのテクニカルライティング入門講座 (高橋慈子著、翔泳社) 2018

読みやすさ: ★★★★★ (非常に読みやすい)

概要: より伝わる文章の書き方のコツ、読みやすさを高めるポイントやユーザマニュアル、提案書、障害報告書の書き方など、相手に伝わる技術文書を効率よく書くための方法を解説しています。

感想: 一文一義を心掛け、読み手が知りたい情報を最初を書くといった技術文書の書き方を学ぶことで一気に読みやすくなります。自分が書いた文章に対して「これどういうこと?」とよく質問される人は、ドキュメントライティングの基礎を学ぶことをおすすめします。

URL: <https://www.shoeisha.co.jp/book/detail/9784798157191>

【書籍】

エンジニアのためのドキュメントライティング (岩瀬義昌 (訳)、日本能率協会マネジメントセンター) 2023

読みやすさ: ★★★★★ (読みやすい)

概要: 新しいサービスの提供を目前に控え、ドキュメント作成に奔走している主人公を描いたストーリー仕立ての書籍です。このストーリーを通じて、開発者向けのドキュメント (README、API リファレンス、チュートリアル、リリースノートなど) の書き方を紹介しています。

感想: プログラマ向けの文章を書く人はぜひ読んでみてほしいです。他の書籍と同じく、基本は読み手が知りたいことを書くことが重要です。

URL: <https://pub.jmam.co.jp/book/b622627.html>

3) 専門家不在のセキュリティレビュー

このページでは、レビューに関するアンチパターンを紹介します。レビューとは、「プロセスや製品の品質を評価し、改善のための機会を識別する活動」です。その目的は「修正工数の低減」であり、プロジェクトの関係者が集まってドキュメント（仕様書、設計書、テスト計画書等）をチェックします。今回のアンチパターンから、もう一度良いレビューについて考えてみてください。

事例

システム設計者の A さんは、重要インフラシステムの設計を担当していました。A さんは設計歴 10 年の中堅ではありますが、セキュリティについては詳しくありません。ある案件を担当することになりました。納期が多少厳しかったですが、それ以外には特筆すべき点の少ない一般的な案件でした。しかし、設計を開始してみると、初めて設計段階でいくつかのセキュリティ上の課題に直面し、不安を感じていました。しかし、さらに相談できるセキュリティ専門家が身近におらず、またプロジェクトのスケジュールは非常にタイトであったため、専門家に相談することなく設計を進めました。設計書には、セキュリティ関連項目として、「セキュリティ対策製品を導入する」「ファイアウォールを設定する」など一通り記載したものの、その内容が実質的に役立つかどうかまでは考慮できませんでした。

A さんが作成した設計書のレビュー会議には、A さん以外に、リーダー（進行役）、レビュー 3 名が参加しました。しかし彼らも A さんと同じくセキュリティに詳しくなかったため、セキュリティに関する指摘はできませんでした。

その後テスト段階に進みました。システムテスト、侵入テスト、受入れテストの順でテストが控えていましたが、侵入テストにて重大な問題が発覚しました。その問題の結果、開発には手戻りが発生してしまい、顧客への納品に大幅な遅延が発生してしまいました。



ヒント：レビュー会議に参加していた、A さん（設計ドキュメント作成者）、リーダー（進行役）、レビュー 3 名は、どのようなことを実施すべきでしたか？

解説

セキュリティの専門家が不在なままドキュメントレビューが進行し、プロジェクトが進むと、重大な脆弱性が見落とされる可能性があります。そして、システムがリリースされた後、攻撃者によってその脆弱性が悪用されると情報漏洩やシステム停止といったインシデントが発生します。これにより企業の信頼性と利益に大きな打撃を与える可能性があります。

Aさん（ドキュメント作成者）やレビュー3名は自分たちがセキュリティに関する十分な知識を持っていないことをリーダーに伝えるべきでした。また、リーダーはレビュー3名のスキルを踏まえ、セキュリティの専門家を召集する必要がありました。

教訓：レビューには、必要なスキルを持つ人をアサイン（招集）する。

対策

それでは、Aさん（設計書作成者）、リーダー（進行役）、レビューは、それぞれどのようなことを実施すべきだったのでしょうか？


- ✓ Aさん（設計書作成者）は、セキュリティに関する疑問点をわからないまま放置したり、レビュー会議中にレビュー達に質問したりするのではなく、すぐにセキュリティの専門家に相談すべきでした。正式レビューの前に、簡易レビューを開催することもよい手段です。
- ✓ リーダー（進行役）は、レビュー準備段階で、適切なレビューを選定し、役割を割り当てる必要がありました。各レビューの専門知識・スキルを理解し、セキュリティレビューを選出しましょう。組織内にそのような専門家がいればよいですが、もし社内にはいない場合は、外部の専門家やコンサルタントに依頼することも検討しましょう。
- ✓ レビュー
 - ◇ セキュリティに詳しいレビューは、設計書作成者に対し、セキュリティの脅威や潜在的な影響を理解しやすく説明します。また、問題の軽減策を複数提示し、それぞれの長所と短所を説明します。これにより、設計書作成者が適切な対策を講じるサポートをします。
 - ◇ セキュリティに詳しくないレビューであれば、ぜひ、セキュリティレビューの指摘内容を聞いてセキュリティについて学んでください。これにより、設計に関する新たな視点を得ることができます。

参考文献


今回のアンチパターンでは「レビューには、必要なスキルを持つ人を召集する」という教訓を得ました。

レビューに人員や時間をかけていても、必要なスキルを持った人がいなければ、重大な問題を見逃してしまう可能性があります。社内にセキュリティについて詳しい人がいる場合、まずはその人に相談してみましょう。専門知識を持つ人の意見や助言を得ることで、設計の品質を向上させ、潜在的な脆弱性を効果的に防ぐことができます。セキュリティは複雑で専門的な分野なので、適切なサポートを受けることが重要です。

生成AI に きいて みた！ #3



生成AI



迷える
初学者

セキュリティレビューのスキルを上げるにはどうすればよいですか？

セキュリティレビューのスキルを上げるには、最新のセキュリティ知識の継続的な学習、実践経験の積み重ね、専門的なトレーニングや認定資格の取得、フィードバックの活用、セキュリティツールの活用、コミュニティとの交流、実際の攻撃シナリオの研究が重要です。

※ すべての生成AIで同じ結果が得られるとは限りません。

セキュリティレビューのスキルを上げるには、レビューがセキュリティの知識と技術を獲得・向上させる方法があります。

- ✓ 知識を増やす
 - ◇ セキュリティ関連のニュースサイトやブログをチェックし、SNS をフォローする。T 社内でのコミュニケーションツールを活用した情報共有も有効です。専門書籍で学び、CISSP や CEH などのセキュリティ認定資格を目指しましょう。
- ✓ 技術を磨く
 - ◇ オンラインのトレーニングコースやハンズオン演習があるイベントに参加しましょう。

これにより、常に最新のセキュリティ知識を身につけ、実務に活かすことができます。

他には、レビューのレビュー自体のスキルを向上させる方法があります。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【書籍】

なぜ重大な問題を見逃すのか？間違いだらけの設計レビュー改訂版（森崎修司、日経 SYSTEMS） 2015

読みやすさ：★★★★★（非常に読みやすい）

概要：この書籍はレビューの正しい心構えを持ち、レビューの正しい手順に従う重要性を伝えることを目的とした文書です。設ドキュメントの「どこ」を「どのように」調べるかを定める「シナリオ作成」や設計ドキュメントのチェック手順（漏れ・曖昧さ・誤り）などについて詳しく解説されています。

感想：セキュリティレビューについて直接書かれていませんが、レビューにおける開発現場の課題とその解決方法について詳しく説明されています。レビューの質を向上に関して課題を感じている方はぜひ、読んでみてください。

URL：<https://bookplus.nikkei.com/atcl/catalog/15/245560/>

【書籍】

セキュアなソフトウェアの設計と開発 脅威モデリングに基づく普遍的アプローチ（ローレン・コンフェルダ、秀和システム） 2023

読みやすさ：★★★★★（非常に読みやすい）

概要：ソフトウェアセキュリティの設計・実装に必要な基本概念やプロセスが記載されています。

感想：ソフトウェアの本ですが、実装以外の部分は、システム設計全般に通ずるものがあります。本アンチパターンに関連するところでは、設計段階で実施すべきセキュリティレビューのプロセスが詳しく解説されています（Chapter 7）。セキュリティレビュー実施時のポイントとして、攻撃者の視点でドキュメントを読んでみようということが書いてあります。

URL：<https://www.shuwasystem.co.jp/book/9784798069753.html>

【WEB サイト】

効果的なペネトレーションテストの始め方～攻撃シナリオを作るコツ（LAC）

読みやすさ：★★★★☆（読みやすい）

概要：このサイトは、効果的なペネトレーションテストの始め方の解説を目的とした文書です。ペネトレーションテストのスコープ、起点、目標、攻撃手法について、どのように考え、設定すればよいかを詳細に説明し、企業が潜在的な脅威を効果的に洗い出すための方法を提供しています。

感想：開発チームが主体となってペネトレーションテストを実施することは一般的ではないかもしれませんが、システムに侵入する側の視点を理解しておくことは非常に有益です。企業によっては、ペネトレーションテストを必須とするところもありますので、その重要性を認識しておくことが大切です。

URL：https://www.lac.co.jp/lacwatch/service/20230412_003342.html

4) 権限設定の見落とし

このページでは、権限設定に関するアンチパターンを紹介します。サイバーセキュリティ対策における「AAA」という言葉を知っていますか？(1)認証 (Authentication) によりユーザの身元確認を行い、(2)認可 (Authorization) によってユーザに対して適切なアクセス権限を付与し、(3)監査 (Accounting) によってユーザの行動やアクセス履歴をチェックすることで、包括的な保護を実現することができます。

事例

TechGrid ソリューションズ社（以下、TechGrid 社）は、重要インフラの制御システムを開発・製造している企業で、世界中の電力・水道施設に技術を提供しています。最近開発をはじめた新しい制御システムでは、セキュリティの強化に向けて多要素認証を導入するなど、認証プロセスの強化に注力していました。その後、新制御システムは完成し、クライアントの電力会社に無事導入されました。

しかし、その数ヶ月後、制御システムの異常動作が報告されました。調査をしてみると、開発段階にエンジニアがテスト目的で作成した管理者権限のアカウントをそのままにしてしまっていたことが発覚しました。そのエンジニア用アカウントから、本来アクセス可能であってはならない運用データや制御機能にアクセスでき、不適切な変更が行われてしまいました。

TechGrid 社はこの問題を受けて全システムの権限設定を見直すことになり、結果として大きなコストを払うことになりました。



ヒント：認証・認可・監査のどこに問題があるのでしょうか？

解説

今回は、新規システムの開発において多要素認証などを導入して認証プロセスを強化したものの、認可設計が適切でなかったために、製品出荷後にも不適切なアクセス権が残ってしまった例を示しました。

具体的には、失敗の原因はシステムの認可設計の不備によるものでした。認証は厳格に行われていましたが、認可が疎かにされており、ユーザごとのアクセス権限が適切に設定されていなかったのです。エンジニア用アカウントの一部が誤って高い権限を保持し続けており、それがこのインシデントの直接的な原因となりました。

不適切なアクセス権は、本来アクセス権がないユーザが機密情報にアクセスできてしまったり、制御システムに対して権限がない作業を実施できてしまったりします。不正アクセスやデータ漏洩などのセキュリティ事故を誘発する恐れがあるため、完全になくす必要があります。

また、今回のアンチパターンでは直接の言及はありませんが、不正アクセスの事例において監査（Accounting）の仕組みが提供され適切に運用されていない場合、セキュリティ事故の検知や原因究明が困難になるケースが想定されます。皆さんは「AAA」の3要素に関して、そのすべての重要性を理解できましたか？

教訓：認証・認可・監査のすべてがセキュリティ対策には不可欠！

対策

認可機能を強化するためには、どうすればよいでしょうか。

認可、すなわちアクセス権を管理する方法として、ロールベースのアクセス制御（Role Based Access Control、RBAC）の導入が挙げられます。RBACにおいて、ユーザは、1つまたは複数の役割を割り当てられ、その役割に応じてシステムへのアクセス権が付与されます。その結果、「多数のユーザに対して一括でアクセス権限を設定できる」「権限管理の一貫性を保ちやすくできる」「ユーザが必要以上のアクセス権を持たないようにすることで最小権限の原則を適用できる」という利点が生まれます。

そして、システム導入時または運用中における定期的な監査によって、適切なアクセス権が設定されているかをチェックしましょう。この結果ユーザの不審な行動や不適切なアクセスを検知できれば、システムの健全性向上につながります。

ちなみに、アンチパターンでは認可に注目しましたが、セキュリティ機能の開発では、認証の仕組みの設計ミスにも気をつけましょう。特に、「認証情報の保護が不十分」「ハードコードされた認証情報の使用」といった不適切な設計に注意する必要があります。


参考文献

今回のアンチパターンでは「アカウント管理には認証・認可・監査のすべてが不可欠である」という教訓を得ました。


皆さんが構築するシステムに認証・認可・監査の仕組みを導入しようとする、「仕組みが複雑で設計やテストに多くのリソースが必要になる」、「セキュリティ要件や規制の変化、また新しい脅威や攻撃手法に対応するために継続的な学習が必要」、「そもそも、セキュリティ標準やベストプラクティスがよくわからない」という方もいるでしょう。

一般的な対策としては、継続的な教育とトレーニングを受ける、最新のセキュリティ技術やベストプラクティスに関する情報を常にアップデートする、必要に応じて専門家のアドバイスを受けることが重要です。もちろん、開発プロセスにおいてセキュリティを最優先に考える「セキュリティ・バイ・デザイン」のアプローチを採用することも推奨されます！

生成AI に きいて みた！ #4



生成AI



迷える
初学者

適切なアカウント管理を行うには
どうすればいいですか？

適切なアカウント管理には、まずユーザー登録と削除を
管理し、強力なパスワードポリシーを適用します。
定期的にパスワード変更を促し、
二要素認証を導入します。
また、アカウントの不審な活動を監視し、
必要に応じてアカウントをロックすることも重要です。

※ すべての生成AIで同じ結果が得られるとは限りません。

まず、アカウント管理の仕組みを一から作るのは大変ですし、作成した機能にバグがあると脆弱性になりかねません。そのため、既に世の中で使われているソリューションやフレームワークを利用できる場合は、それらの採用を検討しましょう。

そのうえで、認証を強化したい場合には必要に応じて通常の ID/パスワードによる認証に SMS 認証/認証アプリケーション/生体認証などを加えた二要素認証の利用を考慮します。また、生成 AI の回答では触れられていませんが（よくあることですね！）、認可すなわちアカウントの権限管理については、やはり RBAC の導入がベストプラクティスなるケースが多いでしょう。

そして監査については、現場での運用方法をよく理解したうえで設計をすることが大切です。アカウントの利用時刻や平常時と異なる活動を AI によって監視・検知する仕組みを有するものも登場しています。監査のユースケースについてシステム運用者と意識合わせをして、適切な設計や設定をすることを意識しましょう。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【書籍】

上流工程でシステムの脅威を排除する セキュリティ設計実践ノウハウ（山口 雅史 IPA） 2018

読みやすさ：★★★★☆（読みやすい）

概要：本書は、上流工程できちんとセキュリティ設計を実施するための実践的なノウハウを、セキュリティの専門家である著者が、詳しく分かりやすく解説します（日経 BOOK PLUS より引用）

感想：「第一章 上流工程で作り込むセキュリティ設計の進め方」の「1-4 アクセス制御と認証の設計」に「IAM」（Identity and Access Management）に関して説明しています。基本情報技術者試験の合格程度の知識があると読みやすいと思います。

URL：<https://bookplus.nikkei.com/atcl/catalog/18/271110/>

【書籍】

体系的に学ぶ 安全な Web アプリケーションの作り方 第2版（徳丸浩著、ソフトバンククリエイティブ） 2018

読みやすさ：★★★★☆（読みやすい）

概要：セキュアなウェブサイトを作りたいウェブアプリケーション開発者（プログラマ）向けの書籍です。Web アプリケーションに生じうる脆弱性と、それを生まないための対策について豊富に、そしてわかりやすく解説されています。

感想：ウェブサイトのセキュリティ対策を主題としている本ですが、情報セキュリティの入門書としても比較的読みやすい内容です。脆弱性や攻撃手法の説明も丁寧に書かれています。が、現在（2024 年）では古くなっている情報もありますので、その点は各人で最新の情報をキャッチアップする必要があります。また、ボリュームが 700 ページ弱のため根気は必要かもしれません。

URL：<https://www.sbcr.jp/product/4797393163/>

【WEB サイト】

安全なウェブサイトの作り方（IPA）

読みやすさ：★★★★☆（普通）

概要：この資料は、ウェブサイトのセキュリティ向上を目的とした文書です。安全なウェブサイトを作成するには、SQL インジェクション対策（1.1 節）や OS コマンドインジェクション対策（1.2 節）など様々な対策を実践する必要があります。1.11 節では、「アクセス制御」や「認可制御」等の機能欠落に伴う脆弱性についての対策が解説されています。

感想：認可機能に限らず、ウェブサイトのセキュリティ対策全般について記載されています。ただし、専門的な用語も少なからず使われており、ウェブや情報セキュリティに関して一定程度の前提知識が必要な資料です。

URL：<https://www.ipa.go.jp/security/vuln/websecurity/index.html>

5) 思いがけない内部犯

このページでは、システム開発時のセキュリティ対策として内部犯への対策をどのようにすべきかを考えます。一般的に想定する内部犯は、組織内の悪意を持った人物だと思います。そのような内部犯への対策には、入退室管理などによる物理的な対策、監査などによる内部犯行に対する抑止、または業務継続計画による早期復旧への準備などが考えられます。それでは、内部犯対策の問題事例を見てみましょう。

事例

重要インフラのシステム設計者である C さんは、ある日、顧客から「内部犯に対応できるシステムにしてください」という要望のある設計を受けました。C さんはすぐにこれに取り組み、システムユーザの内部犯対策を検討しました。彼は、アクセス制御の強化、ユーザ行動の監視システム、ユーザデータマスキング（機密データを実際の値ではなくマスクされた値で表示するように設計する）の 3 つの要素を中心にシステムを設計しました。

数ヶ月後、システムは完成し客先で稼働を始めました。しかし、その数週間後、システムは突然停止しました。調査の結果、システム停止の原因はサプライチェーンのベンダ（協力会社）のスタッフがシステムに不正なコードを埋め込んだことだと判明しました。このスタッフは、システムの脆弱性を利用してシステムを停止させ、ランサムウェアを仕掛けることができました。

システム停止により顧客は大きな損失を被り、事件は大きなニュースとなったため、顧客および C さんの会社は信頼を大きく失いました。また、C さんは、サプライチェーンのベンダに対するセキュリティ対策の重要性を痛感しました。



ヒント：内部犯の想定対象に抜けがあります。

解説

今回の重要インフラシステムで「内部犯」となる可能性があるのは誰なのか考えてみましょう。このアンチパターンで内部犯に該当するのは、システムユーザ（運用者）、システムベンダ（設計・開発者）、サプライチェーンのベンダ（協力会社）です。

このアンチパターンは、セキュリティ対策を考える際には、システムユーザだけでなく、システムベンダ自身やサプライチェーンのベンダも考慮に入れる必要があることを教えてくれます。内部犯対策は、システムの全体的なセキュリティを確保するための重要な要素であり、そこに漏れがあると大きなリスクを生む可能性があります。Cさんが、サプライチェーンのベンダへの対策の検討を漏らしてしまった原因としては、以下が考えられます：

1. 過剰な信頼：
 - サプライチェーンのベンダの関係者を信頼しすぎ、リスクを過小評価した。
2. 知識の不足：
 - サプライチェーンのベンダに対するセキュリティに関する知識や経験が不足しており、その重要性を十分に認識していなかった。
3. リスク評価の不足：
 - リスク評価プロセスが不十分であり、サプライチェーンのベンダに関連するリスクが適切に評価されていなかった。
4. セキュリティの範囲の限定：
 - セキュリティ対策をシステム利用者に限定して考え、システム開発の関係者に対する対策を検討する範囲から漏らしてしまった。

教訓：内部犯の考慮にはサプライチェーンの関係者も含める。

対策

それでは、設計・開発者にできる内部犯対策を考えます。

1. AAA の適用
 - 「4) 4) 権限設定の見落とし」のアンチパターンでも登場しましたが、AAA すなわち認証・認可・監査の適用は、内部犯対策にも有効です。
2. データ損失防止（Data Loss Prevention, DLP）
 - データ損失防止技術を導入することで、機密データの不正な持ち出しや送信を防ぎます。DLP ソリューションは、機密データの転送やコピーを監視し、違反が検出された場合にはアラートを発するか、自動的にブロックします。
3. 最小権限の原則
 - 各ユーザには、その業務に必要最低限の権限のみを付与することで、不要な権限の付与を防ぎます。これにより、内部犯が悪用できる範囲を制限し、被害を最小限に抑えることが期待されます。
4. 多要素認証（Multi Factor Authentication, MFA）
 - システムへのアクセス時に多要素認証を実装することで、システムの他の利用者

へのパスワード漏洩等による不正アクセスを防止します。


また、システム運用者側の対策として、内部監査の強化や、内部犯を想定した侵入テスト／レッドチーム演習なども考えられます。顧客がこのような対策を実施するために必要な機能をシステムに具備しておく必要がないか、お客様との契約前に意識合わせしておくことも大切です。

参考文献

今回のアンチパターンでは「サプライチェーンの関係者も内部犯の考慮に含める」という教訓を得ました。


一般的にはシステムの運用者（システムを内部から制御できる人）を内部犯として想定することが多いと思いますが、最近では内部犯がサプライチェーンに潜んでいる可能性も考えなければなりません。サプライチェーンのベンダに対する評価や監査の実施、納品物のセキュリティ検証やテストの実施および脆弱性スキャンを行う、などの対策も検討してもよいでしょう。

生成AI に きいて みた！ #5



生成AI

内部犯を防ぐにはどうしたらいいですか？



迷える
初学者

内部犯を防ぐには以下の対策が有効です。最小限のアクセス権を与え、多要素認証を導入します。ログを詳細に記録し、定期的に監査します。データは分類して暗号化し、従業員にはセキュリティ教育を実施します。匿名で通報できる制度を整備し、採用時には従業員の背景をチェックします。重要な業務は分散させ、一人で全てを管理できないようにします。これらの対策で内部不正のリスクを減らせます。

※ すべての生成AIで同じ結果が得られるとは限りません。

内部犯対策の基本は、認証の強化と認可(アクセス権)の厳密化です。これにより情報や制御に対する不適切なアクセスを予防します。そして、監査を定期的実施することでアクセス権がある人物の行動をチェックして、適切なアクセスに対しても監視することが重要です。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

[WEB サイト]

組織における内部不正防止ガイドライン (IPA)

読みやすさ：★★★★☆ (読みやすい)

概要：この資料は、内部不正防止の重要性や対策の体制、関連する法律などの概要を説明しています。「基本方針」「資産管理」「技術的管理」「職場環境」「事後対策」等の 10 の観点のもとに、合計 33 項目の具体的な対策を示しています。

感想：インシデントの事例が多数記載されており、自社で起こりうる事象かどうかのチェックリストとして使うことができます。また、内部不正対策の体制例構築例や導入に対する指針が示されており、組織として内部対策に取り組む際のガイドラインとしてよい資料になります。

URL：<https://www.ipa.go.jp/security/guide/insider.html>

[WEB サイト]

2023 年度「内部不正防止対策・体制整備等に関する中小企業等の状況調査」報告書

読みやすさ：★★★★☆ (読みやすい)

概要：企業アンケートとインタビューを通じて、内部不正防止対策の実施状況や課題が収集され、改善策が提案されています。また結論として、内部不正防止ガイドラインの実効性を高めるためには、ガイドラインの概要版の作成や社内規程整備のための内容の増補、従業員への周知・教育に使えるコンテンツの充実などが求められています。

感想：中小企業における内部不正防止対策を、企業に対するヒヤリングをベースにまとめており、内容が具体的な点は読者に受け入れやすいと考えます。ただし文章量が 174 ページと比較的多いので、はじめに「1.概要」および「5.まとめと今後の方向性」に目を通すことをお勧めします。あとは必要に応じて、各章を個別に確認していくのがよいでしょう。

URL：<https://www.ipa.go.jp/security/reports/economics/ts-kanri/20240530.html>

[WEB サイト]

ゼロトラストという戦術の使い方 (IPA) ゼロトラスト移行のすゝめ (IPA)

読みやすさ：★★★★☆ (読みやすい)

概要：「ゼロトラストという戦術の使い方」は、ゼロトラストの基本的な考え方や、導入にあたっての注意点、異なるユースケースでの検証結果などが詳細に記載されています。実際にゼロトラストを導入する際の参考となる内容がまとめられています。一方、「ゼロトラスト移行のすゝめ」は、ゼロトラストモデルへの移行を検討している組織の担当者に向けて書かれており、実際の移行プロセスを支援するためのガイドです。

感想：ゼロトラストは、必ずしも内部犯対策を意図したアプローチではありませんが、情報へのアクセスやシステム操作の際に、その都度認証・認可を徹底するという考え方は内部犯対策にも通じるものです。ゼロトラスト初学者にとっては、ゼロトラスト学習の入り口としても良い資料です。

URL：https://www.ipa.go.jp/jinzai/ics/core_human_resource/final_project/2021/zero-trust.html

URL：https://www.ipa.go.jp/jinzai/ics/core_human_resource/final_project/2022/zero-trust-mgn.html

6) ホワイトリストの登録漏れ

このページでは、マルウェアなど不正プログラムの動作禁止に関するアンチパターンを紹介します。プログラムの起動を制御する手段として、ホワイトリスト（許可リスト）方式のセキュリティ製品を導入する方法があります。この製品は、あらかじめ許可されたプログラムのみが実行できるため、不正プログラムの動作禁止に対して有効ですが、正しく設定しなければ正規のプログラムの動作も阻害してしまいます。また運用中においても、このセキュリティ製品が正しい動作をしているか定期的に検証する必要があります。

事例

重要インフラのシステム設計者である A さんは、顧客である電力会社から「サーバ上で不正なプログラムを動作させないように設計してください」との要望を受けました。A さんはホワイトリスト方式のセキュリティ製品を導入することにしました。

A さんはシステムのテスト環境でホワイトリストを作成するため、すべての正規のプログラムをリストアップしました。しかし、この作業は休日の深夜に行われました。通常業務時間外に行うことで、リストアップ作業の影響を最小限に抑えられると考えたからです。

後日、システムは本番環境で稼働を開始しました。初めは問題なく動作しているように見えたが、深夜にバックアップソフトウェアが稼働を始めました。しかしこのソフトが、ホワイトリストに含まれていなかったため、セキュリティ製品によってバックアップソフトウェアの動作がブロックされました。結果として、定期バックアップが実行されず、データ保全に大きなリスクが生じました。

また、定期メンテナンスのスクリプトも同様にブロックされ、システムのパフォーマンス低下や一部サービスの停止などのトラブルも発生しました。A さんは緊急対応を余儀なくされ、システム全体の点検とホワイトリストの再設定を行うために、多大な時間と労力を費やしました。



ヒント：ホワイトリスト作成時に何があったのでしょうか？

解説

今回は、ホワイトリスト方式のセキュリティ製品を導入した結果、正規のプログラムの動作も阻害されてしまい、正常な運用ができなくなってしまう例を示しました。ホワイトリストを作成する際には、すべての正規プログラムを事前に登録しておく必要があります。

ホワイトリストにプログラムを登録する際には、手動で登録する仕組みと自動で登録する仕組みが想定されます。そして、登録するプログラムの数が膨大な場合、あるいは手動登録における登録ミスを避けるために、多くのケースにおいて自動で登録する仕組みを使うことになるでしょう。ホワイトリストに正規のプログラムを自動で登録する場合、登録対象となるプログラムは、登録処理を実施しているときに対象マシン上で稼働しているものに限られます。裏を返せば、登録処理を実施しているときに稼働していないプログラムは、たとえ正規のプログラムであっても登録されません。

そして、ホワイトリストへの登録処理がこのような仕組みになっていることは、セキュリティ専門家あるいはセキュリティ担当者、またはホワイトリスト登録担当者しか把握していないことがあります。そして、システムの設計者やプログラム開発者がこの仕組みを理解していなかったために、結果的に正規のプログラムやスクリプトの登録漏れが発生してしまうのです。今回は、深夜にはバックアップソフトウェアやメンテナンス用スクリプトなどいくつかのプログラムが稼働していなかったため、これらはホワイトリストに含まれませんでした。

教訓：許可リストはチーム内で共有し、必要なプログラムを漏れなく登録しよう！

対策

ホワイトリストへの登録漏れを防ぐには、どうすればよいでしょうか。ホワイトリスト登録担当者、システム設計者、プログラム開発者それぞれの立場でできることを考えてみましょう：

ホワイトリスト登録担当者

1. 包括的なチェックリスト作成：すべての正規プログラムを網羅する詳細なリストを作成し、定期的に更新します。
2. 定期レビュー：定期的にホワイトリストをレビューし、新しいプログラムや更新されたプログラムが漏れなく登録されているか確認します。
3. 運用監視：実際の運用環境でホワイトリストの適用状況を監視し、ブロックされた正規プログラムがないか確認します。

システム設計者

1. 多段階テスト：システムの異なる時間帯や条件下でテストを実施し、すべての正規プログラムが正しく動作することを確認します。
2. 動作ログの分析：システムの動作ログを分析し、動作していない正規プログラムがないか確認します。
3. バックアップとリカバリプラン：ホワイトリスト設定の誤りに備えて、バックアップとリカバリプランを整備します。

プログラム開発者


1. ドキュメントの提供：正規プログラムの仕様や動作条件を詳細にドキュメント化し、ホワイトリスト登録担当者やシステム設計者と共有します。
2. インベントリ管理：開発したプログラムをリスト化し、アップデートや新規追加の際に迅速にホワイトリストに反映できるよう管理します。
3. 協力体制の確立：ホワイトリスト登録担当者やシステム設計者と緊密に連携し、登録漏れや誤登録が発生しないよう協力します。

このような対策により、ホワイトリストへの登録漏れを防ぎ、安全性を高めることができます。

参考文献

今回のアンチパターンでは「不正プログラムの動作禁止は慎重におこなう」という教訓を得ました。

不正プログラムの動作を停止しようとして、正規のプログラムの動作が止まってしまえば本末転倒です。正規プログラムのホワイトリストへの登録を丁寧に実施するとともに、本番稼働の準備のために仮運用をする期間を設定して、ホワイトリストが期待通りに正しく動作するかを事前に検証することも検討しましょう。



生成AI

生成AI に きいて みた！ #6

不正プログラムを停止するとセキュリティ面でどのようないいことがありますか？

迷える初心者

不正プログラムを停止すると、情報漏洩やデータ破損のリスクが減少し、システムの健全性が保たれます。また、ネットワークやデバイスのリソースが解放され、パフォーマンスが向上します。これにより、システムの信頼性が高まり、サイバー攻撃の被害を防ぐことができます。

※ すべての生成AIで同じ結果が得られるとは限りません。

不正プログラムの検知や停止は、一刻の猶予ありません。最近はランサムウェア被害も増加しており、ビジネスの継続性強化や多層防御の観点からも、確実に取り組みたいセキュリティ対策と考えます。また、ホワイトリストをバイパスする攻撃も世の中には存在します。重要なシステムのセキュリティ対策を検討する際には、こちらも考慮が必要となることを覚えておきましょう。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【WEB サイト】

政府機関等の対策基準策定のためのガイドライン（令和5年度版）（経済産業省）

読みやすさ：★★★★☆（読みやすい）

概要：国の行政機関等のサイバーセキュリティ対策基準として、「政府機関等のサイバーセキュリティ対策のための統一基準群」が策定されました。本ガイドラインは、統一基準の遵守事項を満たすための基本対策事項の例示と、対策基準の策定及び実施に際しての考え方等を解説しています。7.2.2 節に不正プログラム対策の記述があり、具体的な指針や手順が記載されています。

感想：このガイドラインは 530 ページの資料であり、すべてに目を通す必要はありません。まずは 7.2.2 節を読んで、不正プログラム対策の基本的なアプローチを理解しましょう。

URL：<https://www.nisc.go.jp/pdf/policy/general/guider5.pdf>

【WEB サイト】

NIST SP 800-167 Guide to Application Whitelisting（NIST）

読みやすさ：★★★★☆（普通）

概要：この文書は、組織がアプリケーションホワイトリストの基本を理解し、その実装の計画の支援を目的としています。主なポイントは以下の通りです：

- ✓ アプリケーションホワイトリスト：承認されたアプリケーションとアプリケーションコンポーネントのリストで、ホスト上で許可されるものを制御します。
- ✓ セキュリティ対策：マルウェアやライセンスされていないソフトウェア、その他の不正なソフトウェアの実行を防ぐために使用されます。
- ✓ 実装の推奨事項：ホストのオペレーティングシステムにすでに組み込まれているアプリケーションホワイトリスト技術の使用を検討することを推奨しています。
- ✓ 計画と実装：アプリケーションホワイトリスト技術の計画と実装は段階的なアプローチで行うべきです。文書は、これらのプロセスを通じて組織を導くガイドラインを提供します。

感想：アプリケーションホワイトリストについて、基本的な情報がまとめられており良い資料です。英語のドキュメントであるため拒否感があるかもしれませんが、文章量は多くありませんので、一度は目を通しておきましょう。

URL：<https://csrc.nist.gov/pubs/sp/800/167/final>

【WEB サイト】

高度サイバー攻撃対処のためのリスク評価等のガイドライン（NISC）

読みやすさ：★★★★☆（普通）

概要：政府機関等が直面するサイバー攻撃に効果的に対応するための具体的な手順と対策を提供します。また、標的型攻撃への対処方法に焦点を当て、その防御策についても詳しく解説しています。

感想：不正プログラム対策には、不正プログラムの侵入を防ぐことや、侵入後の横展開を防ぐなど、様々なアプローチが存在します。同ガイドラインの付属書では、「標的型攻撃への対処のための対策セット」を示しており、ホワイトリスト方式以外の対策について知ることができます。

URL：<https://www.nisc.go.jp/policy/group/general/risk.html>

7) 消えたログ

このページでは、ログに関するアンチパターンを紹介します。ログとは、システムやネットワーク内で発生する事象（イベント）の記録です。その目的は「過去にどのようなイベントが発生したか、現在どのようなイベントが発生しているか、将来にどのようなイベントが発生しそうか」といったイベントを追跡することです。今回のアンチパターンから、もう一度「ログ」について考えてみてください。

事例

開発チームはあるインフラ企業が、効率的かつ安全な運用を目指して新規開発を受注することになりました。システム開発プロジェクトチームは、様々な要件を収集し、その中に「ログを保存すること」という重要な要件が含まれていました。この要件では、ログの保存期間は具体的に指定されていましたが、どこに保存するかについての指示はありませんでした。

開発チームは、コストや運用の簡便さを考慮し、ログをシステム内の1つのサーバにまとめて保存するという方針を選択しました。これにより、ログの一元管理が可能となり、トラブルシューティングや監査時に効率的にアクセスできる環境が整うと考えたのです。

数年後、インフラ企業の従業員が出社した際、システムの動作が停止していることを発見しました。調査すると、システムがランサムウェアによって暗号化されていることが判明しました。システムを早期に再開させるため、取得していたバックアップを用いて復旧作業を急いで行い、2日間で業務を再開することができました。

同時に、今後のために、セキュリティインシデントの原因究明作業を行うことにしました。何が起こったのかを調査するためにシステムログを確認すると、ランサムウェアによりログファイルが暗号化されていました。そのためログを解析することができず、攻撃者がどのような経路で侵入し、何を実行したのかといった内容を追跡できず、根本的な原因がわかりませんでした。



ヒント：あなたの組織では、どのようなログを、どこに保存していますか？

解説

この事例におけるログ管理の問題は「ログの保管場所」です。サーバ内のデータ（ログを含む）がすべてランサムウェアによって暗号化されたため、ログをセキュリティ事故の原因分析に活用することができませんでした。

ログファイルにはシステムの動作やアクセスの記録などが含まれています。攻撃者は自身の活動の痕跡（侵入経路や侵入後の活動）を隠すために、ログの中身の改竄、暗号化、または削除を実施することがあります。

ログの取得と管理において、「どの機器からログを収集し、どこに集約するか。またどれくらいの期間保存するか」は確かに重要です。しかし、それと同じくらい重要なのが、収集したログを適切に保護することです。

教訓：いざという時にログが利用できるように、ログ管理ポリシーを確認しよう！

対策

ログの分析や活用は顧客の責任範囲ですが、開発チームとして、ログに関してセキュリティ・バイ・デザインの観点からどのように貢献できるかを検討してみましょう。

まず、開発チームは「ログ管理ポリシー」に関して顧客に質問しましょう。そこには法的要件や業界のガイドライン、顧客独自のノウハウが含まれているはずです。漏れや矛盾がないかを精査し、開発するシステムがログ管理ポリシーに従うことを確認します。

次に、顧客が「ログを取得する目的」について、開発チームは質問して明確にしましょう。システム管理者、ネットワーク管理者、セキュリティ管理者等、誰がどのような目的でログを収集しているのかを確認します。そしてそのために必要な情報は何かを把握し、ドキュメントに残しておくことが重要です。

さらに、セキュリティ事故事例と攻撃者の行動を学び、セキュリティ対策を理解することで、顧客に対して多角的な提案ができるようになります。

例1) ログの保管に対する提案

ログを 1 か所に保管していると、ランサムウェアによってログが暗号化された際に復旧が困難になります。分散してログを保存していれば、例えば 1 か所のログが暗号化されて読めなくなったとしても、他の分散したログから復元できます。また攻撃者にとって、自身の攻撃時の挙動を追跡されたくなければ、分散されたログすべてを暗号化または削除する必要があり、攻撃の手間が増えことになります。つまり、ログに対する攻撃への対策として、システム外や複数サーバに分散してログを保管することで、攻撃の難易度を上げることができます。

例2) ログメッセージに対する提案


ユーザの ID やパスワード、個人情報などの機密情報がログに保存されており、情報漏洩に繋がる可能性があります。対策として、ログファイルに対して適切なアクセス制御を行うとともに、ログとして出力する情報を再検討すべきだと考えます。

参考文献


今回のアンチパターンでは「いざというときにログが利用できるように、ログ管理ポリシーを確認しよう！」という教訓を得ました。

ログ管理における一般的な問題は、様々な機器から生成される膨大なログデータをどのように分析し、どこに・どれくらいの期間、どのような情報を保存するかなど、考慮すべきことが多数あることです。

生成AI に きいて みた！ #7



生成AI



迷える
初学者

適切なログ管理とはどのようなものですか？

適切なログ管理とは、システムやアプリケーションの動作を記録し、問題発生時のトラブルシューティングやセキュリティ監査を容易にすることです。重要なポイントは、収集するログの種類と内容を明確に定義し、適切な保存期間を設定し、アクセス制御を行うことです。また、定期的なレビューと分析を通じて、異常を早期に検出し対応する体制を整えることが求められます。

※ すべての生成AIで同じ結果が得られるとは限りません。

適切なポリシーと技術を導入することで、効果的なログ管理を実現し、セキュリティの強化と運用の効率化を図ることができます。

セキュリティだけの観点で言えば、ログは「できるだけ詳細に、長い期間保存しておくこと」が理想です。その理由は、攻撃者が長期間にわたってシステムに潜伏していた場合、その活動を特定するために役立ちます。攻撃者によるシステムへのアクセスや操作の記録が残るため、後から追跡することができます。（リアルタイムなログ分析により不審活動を即座に検出することが望ましい）

- ✓ ログインの成功と失敗（短期間に大量のログイン試行が記録されている）
- ✓ 各ユーザが接続を確立および解除した日時と個々のユーザセッションにおいて送受信されたデータの量（攻撃者が外部に不正に送信したデータとその量）
- ✓ サービスの開始と停止（攻撃に用いるサービスの開始、セキュリティ監視サービスの停止）

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【書籍・電子書籍】

Software Design(ソフトウェアデザイン)2023 年 2 月号 (Software Design 編集部)

読みやすさ：★★★★★ (非常に読みやすい)

概要：開発者や運用担当者が知っておいて損はない Linux システムログの出力の仕組み、Web サーバのログ調査テクニック、ロギング設計の基礎知識について解説しています。

感想：ログ管理に関する概要を手早く把握するのに最適な一冊です。ログ管理についてよくわかっていない初心者の方に向けて、わかりやすく説明されています。そのため、初めてログ管理に取り組む方や基礎を学びたい方には非常に有用です。(電子書籍で入手可能。)

URL：<https://gihyo.jp/magazine/SD/archive/2023/202302>

【WEB サイト】

高度サイバー攻撃への対処におけるログの活用と分析方法 1.2 版 (JPCERT/CC)

読みやすさ：★★★★☆ (普通)

概要：システム管理者向けに、サイバー攻撃への備えと効果的な対処の観点から、ログの取得と分析の方法について基本的な考え方を紹介しています。典型的な組織用ネットワークを構成する各機器において取得すべきログや注意事項、サイバー攻撃の痕跡を見つけるためのログ活用方法について詳しく解説されています。

感想：「C&C」という用語に拒否反応が出なくなってから読んでください。セキュリティ用語が多いので、ある程度の学習をしてからの方が理解しやすいと思います。対象読者はシステム管理者ですが、システムを作るのは開発チームですので、ぜひ開発者にも読んでほしいと思います。

URL：https://www.jpcert.or.jp/research/APT-loganalysis_Report_20220510.pdf

※) JPCERT/CC は、日本国内に関するインシデント等の報告の受け付け、対応の支援、発生状況の把握、手口の分析、再発防止のための対策の検討や助言などを、技術的な立場から行なっている組織。

【WEB サイト】

コンピュータセキュリティログ管理ガイド (IPA)

読みやすさ：★★★★☆ (普通)

概要：米国国立標準技術研究所 (NIST) により公開された NIST SP800-92 「Guide to Computer Security Log Management」を情報処理推進機構 (IPA) が日本語訳した文書です。

ログ管理の重要性およびそのベストプラクティスの提供を目的とした文書です。ログにはどのような種類があるのか、ログ収集・保存に関する指針などが詳しく解説されています。

感想：ログ管理について体系的に学習できます。内容は長いですが、ログ管理には多くの作業が伴いますし、基本的にどんなシステムでもログを取る必要があります。ぜひ読んでおいてください。

URL：<https://www.ipa.go.jp/security/reports/oversea/nist/ug65p90000019cp4-att/000025363.pdf>

8) システムバックアップの落とし穴

このページでは、システムバックアップに関するお話をします。システムバックアップの定期取得は、ハードウェア故障、人的ミスや災害の発生時に迅速にシステム復旧・ビジネス継続する観点で重要です。しかし、システムバックアップ自体のセキュリティは考えられているでしょうか。次のアンチパターンは、システムバックアップにおける情報セキュリティの問題を示す事例です。

事例

エネルテック社は、重要インフラシステムの設計と開発を手掛ける企業で、その設計責任者である B さんは顧客の要求に応えるために日々奮闘していました。顧客からの新たな要求は、情報セキュリティの観点からシステムの堅牢性をさらに強化することでした。B さんはシステムバックアップに課題があると考え、以下のバックアップ戦略を策定しました。

1. バックアップデータの暗号化：データの機密性を保つため、すべてのバックアップデータを暗号化する。
2. オフサイトバックアップの実施：自然災害や物理的攻撃からデータを守るため、地理的に離れた場所にもバックアップを保持する。
3. 定期的なバックアップおよびリカバリテストの実施：バックアップの信頼性を確認し、緊急時の復旧をスムーズにするため、定期的なテストを行う。
4. バックアップデータのアクセス管理：データへのアクセスを厳格に制御し、不正アクセスや組織の内部犯行による脅威を防ぐ。
5. バックアップのバージョン管理と保持ポリシーの策定：データの時系列的な整合性を保ち、法律的な要件に対応するため、バージョン管理と保持ポリシーを策定。

B さんは、セキュリティベンダから上記の戦略に適合するバックアップソリューションの提供を受けてシステムを開発し、なんとか完成にこぎつけました。

しかし、システムが顧客のサイトで稼働を始めてから数ヶ月後、攻撃者がバックアップデータを盗み出す事件が発生しました。このバックアップソリューションに脆弱性があったのです。情報漏洩が発生したのちに、異常を検知した IT チームが調査を始めましたが、時既に遅く、顧客情報がダークウェブ上で販売されていました。この事件はエネルテック社の信頼を失墜させ、顧客からの厳しい質問や規制違反による罰金にも直面しました。

ヒント：バックアップ戦略に抜けがあったようです。

解説

この事例におけるシステムバックアップの問題は「バックアップソリューション自体のセキュリティ評価の見過ごし」です。Bさんがセキュリティ評価を怠ってしまった原因として、以下のよう
な要素が考えられます。

1. リソースの不足：
 - 時間や人員、予算の不足から、バックアップソリューション自体のセキュリティ評価を後回しにした可能性があります。
2. 過信と安心感：
 - 導入しているバックアップソリューションが信頼できるものであるという過信があった可能性があります。既存のソリューションに対して深刻な脆弱性が存在するとは考えていなかったのかもしれません。
3. 情報の不足：
 - 最新のセキュリティ脅威や脆弱性に関する情報を十分に収集・更新していなかった可能性があります。これにより、バックアップソリューションの評価の必要性を認識できていなかったのかもしれません。
4. チーム内のコミュニケーション不足：
 - チーム内でセキュリティ評価の重要性が十分に共有されていなかった可能性があります。コミュニケーションの不足により、セキュリティ評価の実施が抜け落ちてしまったのかもしれません。

これらの要因が重なり合い、Bさんはバックアップソリューションのセキュリティ評価を怠る結果となったと推測されます。

教訓：バックアップソリューション自体のセキュリティ評価を忘れずに！

対策

システムバックアップを適切に行うために、私たちは何をすべきでしょうか。今回のアンチパターンに対する直接的な対策は、バックアップソリューション自体の脆弱性評価の実施です。設計や開発、納品前や現地据付時、そしてその後の定期的な脆弱性評価が必要だったと考えられます。

- ✓ また、Bさんが検討したバックアップ戦略：バックアップデータの暗号化
オフサイトバックアップの実施
- ✓ 定期的なバックアップおよびリカバリテストの実施
- ✓ バックアップデータのアクセス管理
- ✓ バックアップのバージョンングと保持ポリシーの策定

に加えて、

- ✓ システムの多層防御、24 時間体制の監視
- ✓ セキュリティ対策に必要な時間、予算、スタッフ等のリソースの適切な配分
- ✓ 教育やトレーニングによるセキュリティ意識の向上


なども、システムバックアップのセキュリティ強化に効果的でしょう。

参考文献

今回のアンチパターンでは「バックアップソリューション自体のセキュリティ評価を忘れずに」という教訓を得ました。


一方で、システムバックアップの管理には、いくつかの一般的な課題があります。バックアップデータは適切に管理しないと増加する一方です。また、本業に影響をしないためにバックアップのスケジュール調整も重要です。さらに、バックアップが正しく行われ、データが一貫していることや、システム障害時に迅速にデータを復旧できるようにすることも重要です。このような課題をすべて管理できて、ようやくバックアップを活かすことができます。

生成AI に きいて みた！ #8



生成AI

バックアップソリューション自体のセキュリティ評価はどのように実施すればいいですか？



迷える
初学者

バックアップソリューションのセキュリティ評価には、次のステップが重要です。暗号化の有無と強度を確認し、アクセス制御が適切かを評価します。定期的なセキュリティ更新が行われているか、監査ログの機能があるかを確認し、バックアップの保存場所が安全であるかを評価します。最後に、定期的なセキュリティテストを実施します。

※ すべての生成AIで同じ結果が得られるとは限りません。

バックアップは、システムに異常が発生した時に復旧する手段として重要な役割を果たします。サイバー攻撃に限らず、自然災害や人的なミスなどでシステムに問題が起きたときにバックアップからの復旧は最後の砦となります。そのため、定期的に安全にバックアップが取得・保存されて、いざという時に確実に復旧できる仕組みとなっている必要があります。皆さんのシステムではバックアップの仕組みがしっかりとできていますか？

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

49

【WEB サイト】

政府機関等の対策基準策定のためのガイドライン（令和5年度版）（NISC）

読みやすさ：★★★★☆（読みやすい）

概要： 国の行政機関等のサイバーセキュリティに関する対策の基準をとして令和5年7月4日に「政府機関等のサイバーセキュリティ対策のための統一基準群」が策定されました。本ガイドラインは、統一基準の遵守事項を満たすためにとるべき基本的な対策事項の例示と、対策基準の策定及び実施に際しての考え方等を解説しています。3.1.1(8)節にバックアップに関する記述があり、具体的な指針や手順が記載されています。

感想： このガイドラインは530ページの資料であり、すべてに目を通す必要はありません。まずは3.1.1(8)節を読んで、バックアップの基本的なアプローチを理解しましょう。

URL： <https://www.nisc.go.jp/pdf/policy/general/guider5.pdf>

【WEB サイト】

重要情報を扱うシステムの要求策定ガイド Ver. 1.0（2023年7月）（IPA）

読みやすさ：★★★★☆（読みやすい）

概要： この文書は、重要情報を扱うシステムの構築・調達・運用時に、管理者が非平常時に「自律性」を持ち、また環境が変化した時に対応できる「利便性」を備えた、システムの要求仕様を策定できるようなガイドです。要求仕様の項目の策定には、システムの特性評価 → 問題・リスク／利便性要素の選定 → 必要な対策の選定という流れで進められます。

感想： 具体的な要求仕様の項目として、バックアップに関連するものが同ガイドで登場します。自律性確保（データ）の項目では、A-14：データの冗長性確保、A-15：アーカイブストレージの確保、また自律性確保（運用）の項目では、B-7：運用体制の冗長性確保、そして自律性確保（データセンター・通信）では、E-9：データセンター・通信の冗長性確保、となります。バックアップの目的や対策の詳細内容などを確認できます。

URL： <https://www.ipa.go.jp/digital/kaihatsu/system-youkyu.html>

【WEB サイト】

Protecting Data from Ransomware and Other Data Loss Events: A Guide for Managed Service Providers to Conduct, Maintain, and Test Backup Files（NIST）

読みやすさ：★★★★☆（普通）

概要： この文書は米国国立標準技術研究所（NIST）が作成しており、マネージドサービスプロバイダー（MSP）において安全で効果的なバックアップ計画を立てるためのガイドラインです。

感想： バックアップの計画、実践およびテスト、またバックアップに関連する技術について説明をしています。自社にとって適切なバックアップ方法を検討するための資料として利用できます。

URL： <https://csrc.nist.gov/pubs/other/2020/04/24/protecting-data-from-ransomware-and-other-data-loss/final>, <https://www.nccoe.nist.gov/sites/default/files/legacy-files/msp-protecting-data-extended.pdf>

9) 曖昧なセキュリティ設定がもたらすリスク

このページでは、システム内に存在する製品の設定に関するアンチパターンを紹介します。システム開発では、購入した製品を使用してシステム設計を行うのも、設計の1つの形です。今回のアンチパターンから、製品におけるセキュリティ設定について考えてみてください。

事例

システム開発・導入を専門とするベンダ企業 H SYSTEMS 社は、製造業の M-TEC 社から、新しいシステムの開発と導入を依頼されました。M-TEC 社は業務効率化と情報セキュリティの強化を目指しており、H SYSTEMS 社はその要件を満たすために最適なソリューションを提案しました。

H SYSTEMS 社は、業界標準のセキュリティ機能を搭載した自社製品を組み込んで、システム設計を行いました。導入にあたっては、H SYSTEMS 社がシステムの基本的な導入と初期設定を行い、M-TEC 社の IT 担当者に対して基本的な操作説明を行い、その際に詳細なセキュリティ設定を行うためのドキュメント（ハードニングガイド）を提供しました。

初期設定では、管理者アカウントのパスワードポリシーが緩く設定されていましたが、M-TEC 社の IT 担当者はセキュリティ設定の重要性を十分に理解していなかったため、ハードニングガイドを用いたパスワードポリシーの強化を行いませんでした。

数か月後、M-TEC 社内で定期的に行われるセキュリティ監査が実施されました。監査の結果、システムのいくつかの設定が適切でないことが指摘されました。特に、管理者アカウントのパスワードが弱く設定されていることが重大な問題として浮上しました。

M-TEC 社の IT 部門から緊急連絡を受け、H SYSTEMS 社の導入担当者は直ちに状況を確認し、話し合いを行いました。M-TEC 社の IT 担当者は、「H SYSTEMS 社は導入時に提供する際にセキュリティに関してもう少し丁寧に説明する必要があった」と H SYSTEMS 社に対して強く抗議しました。H SYSTEMS 社は M-TEC 社との信頼関係を損なってしまいました。



ヒント：詳細なセキュリティ設定はどちらがすべきだと思いますか？
また「誰が、どの範囲まで責任をもつか」は明確でしたか？

解説

このアンチパターンは設定ミス（設定漏れ）に関するものでした。このアンチパターンでは、ベンダからユーザに対してハードニングガイドという形で、セキュリティを強化するための追加設定（パスワードポリシーの強化、暗号アルゴリズムの強化等）ベストプラクティスの情報を提供していました。

一般的にデフォルト設定はセキュリティが弱いのですが、多くのユーザがそのまま使用してしまします。攻撃者はインターネット上の製品マニュアルからこれらの設定を把握し、悪用の足掛かりにすることができます。

製品の初期設定時に、ページ数の多いハードニングガイドを実行するのは手間がかかり、正直面倒だと感じたことはありませんか？特別な設定をしなくても、デフォルトの状態でセキュリティが確保されていてほしいですね。『セキュア・バイ・デフォルト』とは、システムやソフトウェアが初期設定の段階でセキュリティが確保されることを意味します。製品やシステムを提供する企業は、セキュリティの専門知識がなくても安全に使用できるように、この考え方を取り入れることが重要です。

教訓：可能な限りセキュア・バイ・デフォルト設定にすること！

対策

責任分界点の明確化

はじめに対処すべき問題は、受注側と発注側の「責任分界点」です。責任分界点とは、当事者が複数人いる際の、各人の責任の境界のことです。これによって、各人がどの範囲まで責任を負うのかが明確になります。もし責任分界点が明確でない場合、当事者間で「相手がやるだろう」と誤解し、結果、誰も対応しないというトラブルが発生してしまいます。

発注側と受注側が明確にコミュニケーションを取った上で責任分界点を文書化しましょう。これによって後々のトラブルを回避できます。発注側がセキュリティについて詳しくない場合、受注側は必要な情報を提供し、双方が納得できるように内容を確認することが重要です。

セキュア・バイ・デフォルト

今回のアンチパターンでは、管理者アカウントのパスワードポリシーの初期設定が緩く、ハードニングガイドを用いて詳細な設定を行う必要がありました。ユーザはパスワードポリシーが設定できただけで安心せず、「それは適切なのか？」を判断できる知識を持つことが重要です。

そもそも、ユーザがわざわざハードニングガイドを用いてセキュリティを強化する必要があるのでしょうか？システムやソフトウェアが初期設定の状態でセキュリティ対策が十分に施されている設計思想、つまり「セキュア・バイ・デフォルト」が求められます。これにより、ユーザは特別な設定や追加のセキュリティ対策を行わなくても、安全に使用できるようになります。


ユーザにとって追加作業は面倒ですし、合理的な理由がなければデフォルトで安全な設定にしておくべきです。

参考文献


今回のアンチパターンでは「可能な限りセキュア・バイ・デフォルト設定にすること」という教訓を得ました。

なぜ「設定」が重要なのかを考えてみましょう。§ 1.1.1 でも登場した、IPA の情報セキュリティ 10 大脅威ⁱⁱにおける組織の脅威によると、6 位「不注意による情報漏えい等の被害」は、システムの設定ミスによる非公開情報の公開です。また、9 位「テレワーク等のニューノーマルな働き方を狙った攻撃」でも、テレワーク用製品の脆弱性や設定ミスが攻撃者に狙われるものです。他には、クラウドコンピューティングの重大脅威パンデミックイレブンⁱⁱⁱでも、脅威の 3 位が「設定ミスと不適切な変更管理」となっています。このように、設定ミスは多くのセキュリティリスクに直結しています。

生成AI に きいて みた！ #9



生成AI



迷える
初学者

そもそもセキュリティ設定ミスを発生させないためにはどうしたらいい？

セキュリティ設定ミスを防ぐには、設定の標準化とテンプレート化、自動化ツールの活用、事前検証、詳細なドキュメント作成、複数人によるレビューが重要です。
セキュアバイデフォルト設定も取り入れましょう。

※ すべての生成AIで同じ結果が得られるとは限りません。

今回のアンチパターンで触れた通り、やはり「セキュア・バイ・デフォルト」は重要なようです。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

ii https://www.ipa.go.jp/security/10threats/nq6ept000000g22h-att/kaisetsu_2024.pdf

iii https://www.cloudsecurityalliance.jp/site/wp-content/uploads/2022/11/top-threats-to-cloud-computing-pandemic-eleven-060622-en_us-ja.pdf Cloud Security Alliance (CSA) が公開している「Top Threats to Cloud Computing - Pandemic Eleven」を CSA ジャパンが、CSA の許可を得て翻訳し、公開するものです。

【WEB サイト】

Secure by Default (NCSC^{iv})

読みやすさ：★★★★☆ (読みやすい)

概要：「セキュリティは最初から製品に組み込む必要がある」「セキュリティはユーザの専門的な技術的知識や暗黙的な行動を要求すべきでない」など、全8つの「セキュア・バイ・デフォルト」原則について紹介しています。

感想：英語で書かれているため、人によっては和訳が必要ですが、ブラウザの翻訳機能を使えば簡単に読めます。簡潔にまとめられているため、サクッと読むことができます。

URL：<https://www.ncsc.gov.uk/information/secure-default>

【WEB サイト】

Shifting the Balance of Cybersecurity Risk: Principles and Approaches for Secure by Design Software (CISA)

読みやすさ：★★★★☆ (読みやすい)

概要：この文書は、ソフトウェアメーカーおよびその顧客への推奨事項を紹介しています。メーカーに対しては、セキュリティ・バイ・デザインの原則を満たすこと、その取り組みを公開文書化することを推奨しています。（「セキュア・バイ・デフォルト」はセキュリティ・バイ・デザインの一つです。）顧客には、セキュリティ・バイ・デザインの原則を理解し、ベンダに対してセキュリティに関する質問をすることを推奨しています。

感想：英語で書かれているため、人によっては和訳が必要となります。しかし、36 ページしかなく、業界での影響力が強いガイドラインですので、ぜひ翻訳ツールを活用して読んでほしい内容です。「すべての新車に（追加料金なしに）シートベルトが含まれているように、（ソフトウェア）製品にもセキュリティ構成を含めます。」という例えはインパクトがあると思いました。

URL：https://www.cisa.gov/sites/default/files/2023-10/SecureByDesign_1025_508c.pdf

^{iv} National Cyber Security Centre。英国の国家サイバーセキュリティセンタ。

10) 不要サービスはどこにある？

このページでは、不要サービスの停止に関するアンチパターンを紹介します。不要サービスの停止は脆弱性発生リスクの低減に繋がるため、システムのセキュリティ向上のための重要な対策の1つです。また、その他の効果としてサービスが稼働する機器のリソースの節約や管理の簡素化など、設備や運用面におけるコスト削減にも繋がります。

事例

重要インフラのシステム設計者である A さんは、顧客から新規インフラシステムのセキュリティを強化するために「不要なサービスの停止」を要求されました。このプロジェクトでは、システムを徹底的に分析し、稼働中のサービスの必要性を評価することが求められました。A さんはプロジェクトチームを結成し、システムアーキテクチャのレビュー、デフォルトで稼働しているサービスの調査、および各サービスの依存関係の確認を行いました。加えて、彼らは各サービスの目的と使用状況に関する詳細なドキュメントを作成しました。

しかし、このプロセス中で大きなミスが発生しました。開発とテストの段階で使用されていた複数のモニタリングおよびデバッグツールが、その便利さから本番環境へと移行されてしまいました。これらのツールは、本来は開発およびテスト目的でしか使用しないように設計されており、適切なセキュリティ対策がなされていなかったため本番環境での使用には適していませんでした。

最終的に、客先にシステムが納入され稼働した後、外部のセキュリティ監査を受けた際に、これらのツールが原因で複数のセキュリティ脆弱性が指摘される事態に至りました。この結果、A さんとプロジェクトチームは緊急対応を余儀なくされ、問題のあるツールは速やかにシステムから削除されました。



ヒント：不要なツールが本番環境でも稼働してしまったのはなぜでしょうか？

解説

今回は、開発とテスト段階で導入されたモニタリングツールおよびデバッグツールが、本番環境でも稼働してしまっていたという事例になります。そもそも、顧客要件に含まれていないツールが本番環境に適用されてしまうことはシステムの品質管理の観点において問題ですが、セキュリティ対策の観点からも重大な問題となります。

そして不要サービスの停止は、様々なセキュリティガイドラインやセキュリティフレームワークにおいて提示される基本的なセキュリティ対策となります。なぜ基本的な対策なのかというと、サービスには定期的に脆弱性が発見されることがあり、そのためサービスが多く稼働しているシステムでは脆弱性が発見される可能性が高くなるためです。不要なサービスを停止することで、脆弱性を狙った攻撃を受けるリスクが低下し、また脆弱性管理の対象数が減少することにより運用コストを低下させることができます。

教訓：顧客にとって不要なサービスやツールは削除しましょう！

対策

不要サービスの停止を抜け漏れなく実施するには、どうすればよいでしょうか。事例にも書かれていますが、

- ✓ システムの全体像を把握するためにシステムのアーキテクチャをレビューする
- ✓ 各サーバやクライアントにおいてデフォルトで稼働しているサービスを調査する
- ✓ 各サービスの依存関係の確認する
- ✓ 各サービスの目的と使用状況に関するドキュメントを作成する


などが考えられます。また、今回の事例に関する対策としては、設計・開発メンバが出荷前検査担当にツールの存在を連携することや、出荷前の検査や納品・設置後の客先における検査を実施することも効果的です。

参考文献

今回のアンチパターンでは「不要サービスの停止はセキュリティリスクと運用コストを下げる」という教訓を得ました。ただし、不要サービス停止には、まず不要サービスを見つける必要があります。そのためには、はじめにシステムのサービス一覧を確認します。Windows であれば「サービス管理ツール」を使用します。Linux であれば 'service -status-all' といったコマンドでサービスの一覧を入手できるでしょう。次に、各サービスの目的を理解します。よくわからないサービスについては公式ドキュメントやインターネット検索などで調査しましょう。

そして、システムにおいて不要なサービスを特定します。テスト環境用のサービスや未使用のプリンタサービスなどが該当するかもしれません。そして、特定したサービスを停止し、また機器の再起動後も自動的に起動しないように設定しましょう。


生成AI に きいて みた！ #10



生成AI

不要なサービスを停止するとセキュリティ面でどのようないいことがありますか？

不要なサービスを停止すると、攻撃対象が減り、システムの脆弱性が少なくなります。また、システムのパフォーマンスが向上し、リソースの無駄遣いを防ぎます。これにより、管理が簡素化され、セキュリティの一貫性が保たれます。



迷える
初学者

※ すべての生成AIで同じ結果が得られるとは限りません。

不要サービスの停止をする際に、注意すべき点を述べます。システム設計や開発の場面では、システムの構成部品が社内のリソースでは完結しない場合もあります。例えば、サプライチェーンのベンダからプログラムなどの提供を受ける場合、そのプログラムがサービスとして動作する、あるいはそのプログラムが他のサービスを必要とするケースもあるでしょう。このような場合に不要なサービスを特定するには、サプライチェーンのベンダの協力が不可欠となります。「不要なサービスを停止する」ということを常日頃から意識しておくことが大切です。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【WEB サイト】

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（経済産業省）

読みやすさ：★★★★☆（読みやすい）

概要：この手引きはセキュリティ検証サービスの高度化を目的とするもので、以下の3点について整理しています。(1)機器のセキュリティ検証において検証サービス事業者が実施すべき事項 (2)より良い検証サービスを受けるために検証依頼者が実施すべき事項及び持つべき知識 (3) 検証サービス事業者・検証依頼者間の適切なコミュニケーションのために二者間で共有すべき情報や留意すべき事項。

感想：不要サービスの停止をするためには、どのようなサービスが稼働しているかを調べる必要があります。そして稼働しているサービスを調べる方法として、セキュリティ検証サービスを利用する方法があります。この手引きは、検証サービス事業者が提供するセキュリティ検証について、検証依頼者が実施すべき事項や持つべき知識を記載しています。今回のアンチパターンの場合、3.4.1 項(設計文書レビュー)、3.4.5 項(ネットワークスキャン)に関連する記述があります。

URL：<https://www.meti.go.jp/press/2021/04/20210419003/20210419003.html>

【WEB サイト】

ASM（Attack Surface Management）導入ガイダンス（経済産業省）

読みやすさ：★★★★☆（普通）

概要：このガイダンスは、外部（インターネット）から把握できる情報を用いて不正侵入経路となりうるポイントを把握する ASM（Attack Surface Management）について、関連する各種ツールやサービスの特徴や活用方法について整理しています。

感想：外部に公開するシステムを設計する場合は、システムに対する ASM の実施を検討すべきです。ASM は、外部からの攻撃面の発見や攻撃面の情報収集、そして攻撃面のリスク評価を定期的 to 実施します。そのため、ASM の仕組みを理解し適切に利用することで、対象システムの不要なサービスを検出できる可能性があります。

URL：<https://www.meti.go.jp/press/2023/05/20230529001/20230529001.html>

11) 品質寄りのリリース判定

このページでは、ソフトウェアのリリース判定^vに関するアンチパターンを紹介します。リリース判定では、ソフトウェアに深刻なバグや脆弱性が存在するかが判定基準となります。「バグ」と「脆弱性」は、どちらの言葉も、現代社会では一般用語となっています。しかし、「バグの一部が脆弱性」だということは、意外と認識されていないのではないのでしょうか。

事例

設計開発部の新人 A さんと品質保証部の新人 B さんは、あるシステム開発案件の「設計担当」と「検証担当」として 2 人でアサインされました。

システム全体の設計段階に差し掛かると、設計担当の A さんは、過去に自社で開発したシステム（過去の遺産）を流用すれば、いくつか追加開発はあるものの顧客要求を満足できることがわかり、新人の A さんでも滞りなく設計を終えることができました。システム全体のテスト設計に着手した B さんも同様に、“過去の遺産”と“その修正”により、滞りなく自身の作業を終えました。

その後、開発も無事完了し、システムテストを実施しました。バグは発生しましたが、優先度の低いバグしかありませんでした。そのバグを記載したバグ票^{vi}は下記です。

症状：ユーザが画面のテキストボックスに不正入力すると、画面にエラーメッセージが表示される。

発生条件：「画面のバグで動作に影響しない」かつ「ユーザが不正入力しなければ発生しない」。

原因調査：本来あるべき入力チェックが実装されていない。

対応優先度：低（リリース後対応） ※バグ票の一部抜粋

このバグはリリース判定で話題に挙がったものの、「発生確率は低い」かつ「もし生じても画面更新で復帰する」という理由から対応優先度が下がり、今後修正する方針となり、リリース上の問題にはなりません。その後、システム納入は無事なされました。

システム利用開始から半年が経ったある日、A さん・B さんのもとにシステムがサイバー攻撃された旨の連絡が入りました。攻撃を分析した人物から 2 人が聞いたところ、攻撃者は先のバグが存在した UI を利用してコマンドインジェクションを仕掛けたとのこと。コマンドインジェクションではシステムコードが実行できてしまいます。攻撃時には他の脆弱性も利用されていますが、被害が甚大であり、コマンドインジェクションの脆弱性をリリース判定時に気が付いて修正していれば攻撃を途中で防げたかもしれないという思いから、2 人はリリース判定の不備を悔いることとなりました。

ヒント：リリース判定では何が悪かったのでしょうか？

^v システムをリリースしてよいか否かを決定する会議。

^{vi} バグ対応内容の記録のこと。バグの症状・発生条件・対応履歴などが記載される。

解説

彼らは上記案件において何がよくなかったのかを振り返ってみました。その結果、根本原因は、リリース判定のバグ対応優先度の判断で、品質面のみに意識が向き、セキュリティ面への配慮が欠けている状態、言い換えると「リリース判定の形骸化」だと考えました。

このアンチパターンが教えてくれるのは、品質面でのバグには、常にセキュリティ観点のリスクが伴うことです（本ケースは少し極端な例ではありますが、インパクトを大きくするためリリース判定を題材としました）。システムにおいて、品質／セキュリティの厳密な区別は難しく、結局、設計・開発における品質向上がセキュリティ向上、つまりセキュリティ・バイ・デザインの第一歩となります。

しかしそうは言っても、「すべてのソフトウェアにバグは存在する」というくらい、バグのないソフトウェアの実現は困難です。「バグがなくならないなら、セキュリティも考慮しなくていいんじゃない？」と考える人もいるかもしれませんが、それではいけません。セキュリティは、設計・実装を行う人間の意識にも大きく根ざしていると言われています（少なくとも人間が設計・開発している現状では）。そのため、設計・開発者の意識が低下すると、簡単にたくさんの意図しない動作を生むことになり、それらの要素が複合的に重なった結果、攻撃者に攻撃する隙を与えることとなります。

教訓：「攻撃者が利用可能なバグ」が脆弱性であり、品質面だけでなくセキュリティ面からもバグを見ないと、リリース判定が形骸化してしまう。

対策

セキュリティ向上には、大前提として、まず品質向上が重要です。担当者ごとに下記点を考慮してください。

設計者/品質保証担当者

- ✓ システムの設計品質を保証するのは、設計者や品質保証担当者です。よくない設計は、後工程の実装に影響を与えます。
- ✓ 設計者や品質保証担当者がセキュリティに詳しくない場合には、専門家に相談しましょう。特にリリース判定では必ず専門家が入るようにしたいですが、手戻りの可能性を考え、事前に相談をしておくとい良いでしょう。


実装者（プログラマ）

- ✓ プログラムの品質向上のため、セキュアコーディングを身につけましょう。
- ✓ セキュアコーディングの学習に時間が掛かる場合には、セキュアコーディングを身につけたプログラマとペアプログラミングを実施するのも有効です。
- ✓ セキュアコーディングを身につけたプログラマにコードレビューをしてもらいましょう。

開発初学者の方々は、品質の欠如がセキュリティの欠如になってしまうという怖れを抱くかもしれませんが、反対に言えば、品質に気をつけることが、攻撃者に付け入る隙を与えないための第一歩となります。

参考文献

今回のアンチパターンでは、「攻撃者が利用可能なバグが脆弱性であり、品質面だけでなくセキュリティ面からもバグを見ないと、リリース判定が形骸化してしまう」という教訓を得ました。リリース判定に限らず、その前の設計・実装時、レビュー時、テスト時などから常に、そのバグがどんな影響を及ぼすかを考えなければいけません。設計・実装時には、上述の「対策」を行いましょ。しかし、設計・実装の規模が大きくなるほどシステムも複雑性を増し、どうしてもバグは埋め込んでしまうものです。



生成AI

生成AI に きいて みた！ #11

製品・システムにバグを埋め込まないためには何をしたらいい？

迷える初心者

製品やシステムにバグを埋め込まないためには、設計段階からテストやコードレビューを徹底することが重要です。明確な要件定義、モジュールごとのユニットテスト、自動化テストの導入、そして定期的なコードレビューを行い、品質を確保しましょう。また、開発者間のコミュニケーションを活発にし、問題が早期に発見される環境を作ることも大切です。

※ すべての生成AIで同じ結果が得られるとは限りません。

色んな観点の回答が返ってきましたね。「定期的なコードレビュー」とありますが、ここでは、定期的にコードを振り返るという意味で「リファクタリング」を取り上げましょう。

リファクタリングは、コードの外部的な機能性を変えずに、内部の複雑性を取り除いてきれいに直す作業です。機能性は変化しないので、プログラマ以外の人からすると、価値を生まないように感じるかもしれませんが、定期的なリファクタリングはシステムの複雑性への有効な対抗手段となります。

まず、複雑なコードは何かあった場合に修正時間がかかります。修正前の原因箇所特定と、修正後の影響範囲特定が難しいからです。しかし、実装では、最初からコードをきれいに書くのは困難で、動かさないとわからないことも多いので、先に動く状態にコードを仕上げることも重要です。そこでリファクタリングの出番です。定期的にコード品質をメンテナンスし、複雑性を低く維持するのです。注意点として、リファクタリングでは機能性を変えてはいけない（間違っても機能低下させてはいけない）という「縛り」があるので、実施前後のテストが重要となる点を述べておきます。自動テストを作っておくのがよいとされています。

以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【ガイドライン】

デジタル・ガバメント推進標準ガイドライン 実践ガイドブック（デジタル庁） 2024

読みやすさ：★★★★☆ (普通)

概要：政府の情報システムの整備・管理で守るべきルールを書いています。一般的なシステムに適用できる内容です。ページ数は多いものの、比較的平易な言葉が使用されています。

感想：「Step 6」の「1 本番移行と本番稼働の開始を承認する (p.355)」にリリース判定に関する言及があり、判定基準をあらかじめ明文化することが重要であると述べています。

URL：https://www.digital.go.jp/assets/contents/node/basic_page/field_ref_resources/e2a06143-ed29-4f1d-9c31-0f06fca67afc/d85eeb55/20240605_resources_standard_guidelines_guideline_05.pdf

【書籍】

セキュアなソフトウェアの設計と開発 脅威モデリングに基づく普遍的アプローチ（ローレン・コンフェルダー、秀和システム） 2023

読みやすさ：★★★★☆ (読みやすい)

概要：ソフトウェアセキュリティの設計・実装に必要な基本概念やプロセスが記載されています。

感想：「Part」「Part 3 実装」では、脆弱性や、その対策となるコーディング方法（セキュアコーディング）について書かれています。インジェクション攻撃に関する記述もあります。

URL：<https://www.shuwasystem.co.jp/book/9784798069753.html>

【書籍】

セキュア・バイ・デザイン 安全なソフトウェア設計（マイナビ） 2021

読みやすさ：★★★★☆ (やや読みづらい)

概要：セキュリティ向上のために、プログラミング品質を向上させるという、ソフトウェアのセキュア・バイ・デザインの本。最近普及している「ドメイン駆動設計」という設計上の考え方を大きく取り入れている。

感想：いわゆるセキュアコーディングに関して書かれた一冊です。オブジェクト指向プログラミングなどのプログラムの基礎知識やドメイン駆動設計の基礎知識を身に付けた状態で読むことが推奨されます。

URL：<https://book.mynavi.jp/ec/products/detail/id=124056>

12) 自作コードへの過信

このページのアンチパターンはいわゆる「車輪の再発明」に対する注意喚起です。一般的に、自作コードでは、フレームワークやライブラリに比べセキュリティ機能が不足します。つまり、脆弱性がシステムに埋め込まれる可能性が上がります。

事例

アプリケーション開発担当 Aさんは、システム開発案件にて、システム操作画面を設計・実装することになりました。

設計に手を付けはじめた彼は、まず操作画面を「WEB アプリ」で実現することを決めました。WEB 開発のフレームワーク^{vii}には、社内の過去案件と同じものを使うことを決定しました。そうすることが、ユーザの要求する画面デザインを満足しつつ最も素早く実装できそうだったからです。

続いて、簡単な事前調査を行うことで、本当にそのフレームワークで顧客が望む要求を満足できるかを確認しました。すると、画面の基本的な部品（テキストボックス、プルダウンメニュー等）は、フレームワークの標準機能で実現できそうです。しかし、一部の部品は、フレームワークに用意された機能でカバーできないようです。

フレームワーク標準機能を利用するだけの実装は面白くないと考えていた彼は、その部品に関しては、自作で実装することにしました。まず彼は、WEB 検索から参考となるコードを探しました。そのコードを少し修正すれば、それほど工数を割かずに実装可能と結論づけました。

無事に実装および設計者検証を完了した数週間後、同案件に参加している品質保証部の同僚から連絡が来ました。脆弱性が発見されたようです。

詳細をきくと、その脆弱性は彼が自ら実装した箇所に埋め込まれていました。脆弱性が結合テストの段階で納品前に発見されたことは幸いでしたが、開発工程の手戻りに加え、自作した WEB パーツはどれも修正が必要なため修正工数は大きく、納期遅れが発生してしまいました。



ヒント：自作コードとフレームワークの違いは何でしょうか？

^{vii} プログラマの開発を簡便にするために、よく使用する機能をまとめた開発用ライブラリ。

解説

彼は上記案件で何がよくなかったのかを振り返ってみました。彼は、自身の専門知識や関心がコーディングに偏っており、WEB アプリケーションのセキュリティ対策に関する十分な知識がないことに気が付きました。

セキュリティ対策には、専門知識と枯れた技術^{viii}が欠かせません。上記のアンチパターンは、セキュリティに明るくないプログラマが自作コードを作る際の危険性を示しています。また、仮にセキュリティ知識に自身を持っていても、様々な人に長く使われたフレームワークと同等のセキュリティ品質を作り上げることは難しいと認識することも大事です。

教訓：自作コードの作成は極力避けましょう。

対策

プログラマは下記の点に注意しましょう。


- ✓ 開発言語・開発環境・フレームワークのセキュリティ機能を理解し、開発に何を使うのかを適切に決定しましょう。
 - ✧ 例えば、本アンチパターンのAさんは、開発にあるフレームワークを利用しました。しかし、その決定基準は、あくまで開発効率であり、彼はセキュリティ観点でもフレームワーク利用が重要であることを理解していませんでした。フレームワークやライブラリは、多くの専門家によって検証されており、自作するよりも安全である場合が多いです。一方で、セキュリティ機能が存在しても、それを適切に使用できなければ意味がありません。使い方を調べ、正しく使用できるようにしましょう。
- ✓ フレームワーク利用する場合、使用できる機能が限られ、顧客要求を満たすことが難しい場合もあるかもしれません。そのような場合には、自身でセキュアな機能実装やテスト設計・実施をしなければならないことでしょう（実装は素早くできても、テスト工数がばかにならないということも往々にしてあります）。それを見越し、必要な工数や予算を早期段階で確保すべきです。
- ✓ 自作コードだけでなく、「WEB 公開されているコード」や「オープンソースソフトウェア（OSS）」、「社内過去案件のコード」など、既存コードの使いまわしにも気をつけてください。また、それらはバージョンによって実装状況が変わるので、バージョンにも注意しましょう。
- ✓ 開発環境・フレームワークに備わっている静的コード解析などのツールも利用するようにしましょう。よくあるミスを確認する作業は、人よりもコンピュータが得意です。ただし、利用するだけで、セキュリティが保証される訳でないことには注意をしましょう。

これらを大局的に捉えると、結局「ありものを有効に使えるよう、使用前にきちんとした調査が必要」ということに収斂されるのかもしれません。

^{viii} ここで言う「枯れた技術」は、古い技術という意味ではなく、「広く使われることで十分信頼性が高くなった技術」を指します。「経験豊富な」という認識が近いと思います。


参考文献

今回のアンチパターンでは「自作コードの作成は極力避けましょう」という教訓を得ました。しかし、使用するフレームワークでは実現できない要求（古い環境を使用しなければならない、古いコードを使用しなければならないなど）が顧客から来た場合に、独自で対応しなければいけないのもまた事実です。そのような場合にはどうすればよいのでしょうか。



生成AI

生成AI に きいて みた！ #12



迷える
初学者

自分が書いたコードがセキュアかどうか確認する方法を教えてください。

セキュアなコードか確認するには、次の方法があります：静的解析ツールでコードをスキャン、コードレビューを実施、セキュリティテスト（例：ペネトレーションテスト）を行い、最新のセキュリティベストプラクティスに従うことが重要です。また、定期的なセキュリティトレーニングを受け、脆弱性について最新情報を把握しましょう。

※ すべての生成AIで同じ結果が得られるとは限りません。

出力結果をまとめると、下記のようになりますね。

- ✓ ツールに頼る、またベストプラクティスを取り入れる
- ✓ 定期的に、最新情報を収集して自身の知識・スキルを更新する

後者に関して補足します。あらゆる攻撃に対しセキュアというソフトウェアやシステムを作り上げるのは非常に難しいことですし、今現在セキュアだと信じていても、変化の速い世界ですので、その状況はすぐ変わるかもしれません。そのため、知識・スキルの更新は必須です。

ここで、少し方向性の異なる話をすると、そもそもセキュリティ知識・スキルを持つプログラマーとそうでないプログラマーを区別するような制度が必要なのかもしれません。知識・スキルの更新等の自己研鑽がすべて自己責任（かかるお金も含めて）となると、プログラマーのやる気が削がれるかもしれません。そこで、例えば、通常開発速度などで評価されがちな人事制度を変え、セキュリティ知識・スキルを持つか否かで給料が変わるような人事制度を作ります。準備は大変かもしれませんが、有効な手段に思います。

少し話が逸れましたが、以下に筆者らが生成 AI に質問して得られた結果を踏まえた参考文献を紹介し、読んでみた感想や読みやすさに関する評価を行いました。

【ガイド】

脆弱性対処に向けた製品開発者ガイド (IPA) 2020

読みやすさ：★★★★★ (非常に読みやすい)

概要：製品の開発側で、脆弱性はどう対策すべきかのガイド。

感想：広い内容を含む。方針・組織的な対策から、設計・開発時の方法までが体系立ててまとまっている。セキュアコーディングを学ぶ前に、セキュアコーディングが脆弱性対策の一部だと正しく理解できると感じます。

URL： <https://www.ipa.go.jp/security/guide/vuln/ug65p90000019bum-att/000085024.pdf>

【WEB サイト】

セキュアコーディング (JPCERT コーディネーションセンター)

読みやすさ：★★★★☆ (普通)

概要：セキュアコーディングのための資料やセミナー情報を紹介しています。また、サイト内の「CERT C セキュアコーディングスタンダード」では、体系化されたセキュアコーディングの規則や作法を紹介しています。

感想：セキュアコーディングを身につけたいプログラマが読むとよいと思います。資料には脆弱性とその対策の詳細な説明が記載されており、「CERT C セキュアコーディングスタンダード」は実際にセキュアコーディングを行う際のコーディング規約です。資料は、プログラマ以外も知識として知っておいた方がよいかもしれませんが、コーディング規約の方は、プログラマ以外が読む必要はないと思います。

URL： [ホーム] <https://www.jpcert.or.jp/securecoding/>, [コーディング規約]
<https://www.jpcert.or.jp/sc-rules/00.introduction.html>

【書籍】

体系的に学ぶ 安全な Web アプリケーションの作り方 第2版 (徳丸浩著、ソフトバンククリエイティブ) 2018

読みやすさ：★★★★☆ (読みやすい)

概要：セキュアなウェブサイトを作りたいウェブアプリケーション開発者 (プログラマ) 向けの書籍です。Web アプリケーションに生じる脆弱性と、それを生まないための対策について豊富に、そしてわかりやすく解説されています。

感想：初級～中級者向けとしてよく知られた標準的な一冊です。脆弱性の対策のためのコーディング技術 (セキュアコーディング) はもちろん、それをしないとどうなるかもわかりやすいです。

URL： <https://www.sbcr.jp/product/4797393163/>

第3章 おわりに

ここまで読んでいただき、ありがとうございます。お疲れ様でした！

本書を読み終えた後は、本書での学習内容を踏まえて、ぜひご自身の業務を「セキュリティの観点」から振り返ってみてください。その後、過去の開発プロジェクトに関する資料を見直し、システムが大丈夫か確認してみましょう。そうすることで、業務全体のセキュリティレベルを向上させ、潜在的なリスクを早期に発見することができます。

セキュリティの学習は一度取り組んだだけで終わりではありません。常に進化する脅威に対抗するためには、セキュリティに関して継続的な情報収集やスキルアップが必要不可欠です。本書を読み終えた後にも、学習を続ける方法を付録 A で紹介しています。

さらに、経営層が、セキュリティの重要性を理解し、現場のメンバを積極的にサポートすることで、組織全体に「セキュリティを重要視する意識（セキュリティの文化）」を浸透させることができます。セキュリティの文化を組織に根付かせるための具体的なステップは、付録 B で詳しく説明しています。

これからも一緒に頑張りましょう！

謝辞

本書の作成にあたり、橋本先生、門林先生には、IPA 産業サイバーセキュリティセンター中核人材育成プログラムの講師として、また本プロジェクトのメンタ・副メンタとして、たくさんのご指導・ご助言・ご支援を賜りました。あらためて御礼を申し上げます。

また、不自由なくプロジェクトに取り組める環境を与えてくださった IPA 事務局およびプロジェクトメンバの派遣元企業各社の方々にも御礼を申し上げます。

最後に、同時期に卒業プロジェクトに取り組んだ、中核人材プログラム 7 期生の皆様にもこの場を借りて感謝を申し上げます。

システム開発のセキュリティ向上プロジェクト 0.0

小山 千尋、山中 晋爾、兼子 翔伍

付録 A セキュリティの学習方法

本書はセキュリティ初学者向けであり、セキュリティを学ぶ「最初の一步」を踏み出してもらうことを目的としています。この本のアンチパターンは、セキュアな開発に必要なすべての観点を網羅している訳ではありません。セキュリティ分野は、非常に広範で複雑、かつ変化が速いです。本書を読み終えて「サイバーセキュリティについてもっと知りたい!」と思っただけの読者向けに、「次のもう一步」を踏みだすための方法を紹介します。

学習方法には以下の2つが考えられます。

- ✓ 個人学習: 読書による知識獲得やハンズオン演習による技術習得 等
- ✓ 集団学習: 社内勉強会によるコミュニケーション強化や他業種交流によるネットワーキング活動 等

ここでは、「次のもう一步」を踏み出す方法を具体的に説明します。

個人学習

「個人としてのレベル向上」では、以下の3つの方法を紹介します。

1. 知識習得: 書籍や動画、オンラインコースを利用した学習

セキュリティは非常に多くの要素で成り立っています。そのため、ハードウェア、OS、アプリケーション、プログラミング、ネットワークなどの基礎をしっかりと理解することがとても重要です。

基礎習得には、資格取得が有効です。試験に向けた学習を通じて、体系的に知識を習得できます。同時に、成長を実感することで学習のモチベーション向上にもつながります。

また、有名書籍、動画、オンライン学習プラットフォームの活用もよい方法です。動画やオンライン学習は、視覚的にも理解を深められます。自分に合った学習方法を選びましょう。

2. 情報収集: ニュースサイトやソーシャルメディアの購読・フォロー

流れの速いセキュリティ業界では、ニュースサイトやソーシャルメディアによる情報収取は他分野以上に重要かもしれません。セキュリティ関連のニュースサイトを購読し、セキュリティに関する情報を発信しているソーシャルメディアアカウントをフォローし、またセキュリティコミュニティに参加することで、自然と最新のセキュリティ情報が集まるようになります。

3. 技術習得: ハンズオン演習による実践的なスキルの獲得

理論だけでなく、セキュリティ演習用の環境で手を動かしながら学習することで、実践的なスキルが身に付きます。CTF (Capture The Flag) コンテストや毎日1つのCWEの内容を確認

ⁱ 個人学習・集団学習は相互に関係するため、厳密に区別できるものではありません。

し、検証する等のハンズオン演習により、より技術に対する理解が深まります。

集団学習

個人としてのレベル向上を狙う上でも、「集団の力」を借りるとことで、以下の 2 つの効果が得られます。

1. モチベーションの維持・向上

セキュリティを学習する上でぜひ読んでいただきたい文書の 1 つに『NIST SP800 シリーズ』があります。この文書は、セキュリティ対策に対する網羅性が高く、1 つのトピックに対し少ないもので 50 ページ程度、多いもので 450 ページに及ぶこともあります。これを 1 人で読むのは大変で、途中で挫折してしまうこともあります。しかし、仲間がいれば、その人の経験や知識を共有しながら一緒に読み進めることができます。『早く行きたければひとりで行け、遠くへ行きたければみんなで行け』というアフリカの諺のように、仲間とともに、より深い知識を継続的に学習することができるのです。

2. 知識と多様な視点の共有

T 型人材 (T-shaped person) という言葉を聞いたことがありますか？T 型人材とは、幅広いスキル (知識と技術) を持ちながらも、特定の分野においてより深いスキルを持っている人を指します。個人で獲得した知識や習得した技術を誰かに伝えてみましょう。また異なる専門分野の人の話を聞いてみましょう。対話を通じて幅広い視点を持ち、異なる分野からのインスピレーションを得ることができます。そのような人々とコラボレーションすることで、革新的な解決策を生み出し、幅広い専門スキルを持つ集団を構築できます。各メンバーの強みを最大限に生かすことができるでしょう。

また、わからないことがあれば、積極的に質問してみましょう。相手がわからないことがあれば、自分が知っていることを教えてあげることで、相互に学び合うことができます。

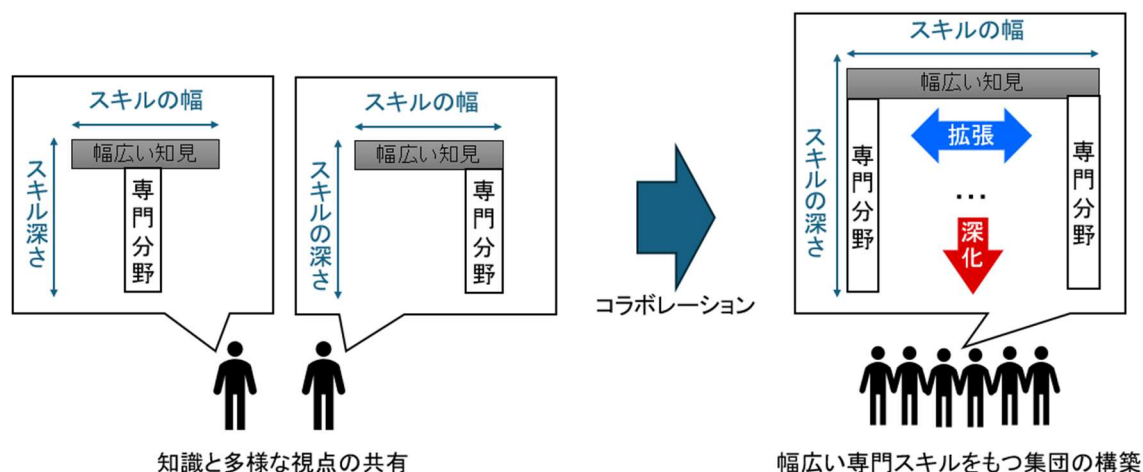


図 A-1: T 型人材の概念図。

最後に、セキュリティはあらゆる場所で必要とされているため、学んだことが無駄になることはありません。すべての学びが必ず役に立ちます。学習を継続してください！

付録 B セキュリティ文化の形成

文献ⁱⁱでは組織のサイバーセキュリティ文化を「組織をサイバー攻撃から保護および防御するために従業員の行動を促す信念、価値観、態度」と定義しています。組織内にサイバーセキュリティの文化を築くことで、従業員が組織のサイバーセキュリティを維持する行動を実践し、セキュリティ事故に対して迅速に対応し、回復する能力を高めることができます。

セキュリティ文化を確立するためには、経営層がサイバーセキュリティの重要性を強調し、その文化を推進するためリーダーシップを発揮することが必要です。以下の図は、文献ⁱⁱⁱを参考にして筆者らが作成したものです。

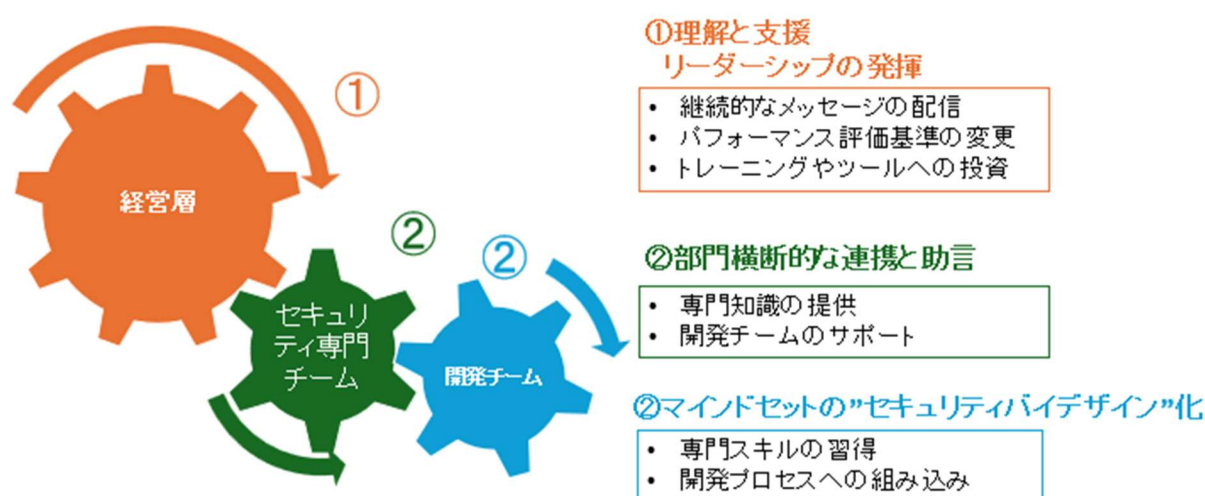


図 B-1: セキュリティ文化の形成。

この図は、経営層のリーダーシップによってセキュリティの文化が形成され、各チームが円滑に連携してセキュリティを組み込んだ製品開発に取り組む様子を表現しています。

経営層

- ✓ 継続的なメッセージの配信：セキュリティの目標や方向性を明確に示し、従業員が共有できるビジョンを伝えます。
- ✓ パフォーマンス評価基準の変更：「製品を早くリリースする人」や「エレガントなデザイ

ⁱⁱ Huang, Keman, and Keri Pearlson. "For what technology can't fix: Building a model of organizational cybersecurity culture." In Proceedings of the 52nd Hawaii International Conference on System Sciences. 2019. <https://web.mit.edu/smadnick/www/wp/2019-02.pdf>

ⁱⁱⁱ Dr. Keman Huang and Dr. Keri Pearlson. "From Bolted-On to Baked-In: Designing Cybersecurity into Digital Offerings". 2021. https://cams.mit.edu/wp-content/uploads/ProdDevProcess_HuangPearlson_Feb102021.pdf

ンにする人」よりも、「セキュリティを設計段階から組み込む人」や「製品やサービスの問題点や脆弱性を発見・修正する人」を高く評価します。これにより従業員の行動を促します。

- ✓ トレーニングやツールへの投資：開発チームはセキュリティを製品に組み込むためには、専門知識の獲得やセキュリティに関する考慮事項の文書化等が必要で、これが負担となります。この負担を解消するために、開発チーム向けのセキュリティトレーニングやツール、セキュリティ専門家に相談できる環境整備といった投資が必要です。これらの投資がないと、セキュリティ文化の持続は困難です。

セキュリティ専門チーム

- ✓ 専門知識の提供：開発チームにセキュリティの知識を提供し、知識のギャップを埋める。
- ✓ 開発チームのサポート：開発チームがセキュリティの問題に直面した時にサポートする。

開発チーム

- ✓ 専門スキルの習得：セキュリティの知識を身につける。
- ✓ 開発プロセスへの組み込み：セキュリティを重視し、設計や実装プロセスにセキュリティを組み込む。

また別の文献^{iv}では、セキュリティ事故の振り返りを行い、学びを得る「ポストモーテム文化」や全員が「自分にもセキュリティに対して責任がある」と自覚を促すための学習方法を検討・実践しています。

セキュリティの文化形成には、経営層のリーダーシップが必要となります。「経営層と話す機会がない、セキュリティの重要性について全く理解してもらえない」という方は、まずは個人で小さくはじめるという方法もあります。こちら^vを参考にしてください。

各企業は異なる文化を持っているため、すべての組織に同じセキュリティ文化を適用することはできません。他社の良い事例を参考にしつつ、自社に適した形に調整することが重要です。

^{iv} <https://www.oreilly.co.jp/books/9784814400256/>

^v <https://kaizenjourney.jp/>



独立行政法人情報処理推進機構(IPA)

産業サイバーセキュリティセンター

第7期中核人材育成プログラム 卒業プロジェクト

