TLP:CLEAR

セキュアバイデザイン

サイバーセキュリティ・リスクの
バランスを変えるために：

セキュアバイデザインなソフトウェアに向けての
原則とアプローチ

TLP:CLEAR

Communications Security Establishment

Canadian Centre for Cyber Security

Centre de la sécurité des télécommunications

Centre canadien pour la cybersécurité

もくじ

はじめに：設計由来の脆弱性

インターネットに繋がったシステムは、経済的繁栄から暮らしと生計、さらには健康にまで直接影響を与えるような重要システムに私たちを結びつけており、その対象範囲が個人識別情報の管理から医療ケアにいたるまで日に日に拡大して行く中で、テクノロジーは私たちの日常生活のほぼすべての面に溶け込んでいる。そうした様々な利便性が抱えるデメリットの一例として、グローバルなサイバー攻撃による情報漏洩が原因で病院での手術が見送られたり、患者ケアが別の医療機関に移管されるという事例が挙げられる。重要システムにおけるセキュアでないテクノロジーや脆弱性は、悪意あるサイバー侵入を招きかねず、潜在的な安全[1]リスクにつながる可能性がある。

このため、ソフトウェアメーカーがセキュアバイデザインかつセキュアバイデフォルトの姿勢を製品設計と開発プロセスの焦点とすることが、極めて重要である。一部のベンダーがソフトウェア保証の面で大きく進歩し、業界を推進させてきた一方で、遅れをとり続けているベンダーもある。本文書の作成に携わった各機関はすべてのテクノロジー製造業者に対して、顧客が自らのシステムについてサイバー侵入を抑えるために常時モニタリングや定期的な更新、ダメージ・コントロールなどを行う必要がないようにするなど、サイバーセキュリティ面での顧客側負担の軽減に基づいて製品を構築するよう強く推奨している。また、我々はソフトウェアメーカーに対し、設定やモニタリング、定期的な更新の自動化を促進させるような方法で製品を作成することを強く求めている。メーカーは、顧客にもたらされるセキュリティ上の結果の改善について、自らの責任で管理するよう推奨されている。従来、ソフトウェアメーカーは顧客が製品を導入した後に発見される脆弱性の修正に依存してきたため、顧客は自ら費用を負担してそれらのパッチを適用しなければならなかった。修正プログラムの作成・適用を延々と繰り返すというこの悪循環を断ち切るための唯一の手段は、セキュアバイデザインの慣行を導入することである。
注：「セキュアバイデザイン」という用語は、セキュアバイデザインとセキュアバイデフォルトの双方を包含している。

この高水準のソフトウェアセキュリティを実現するために、本書作成に携わった各機関はソフトウェアメーカーに対して、製品セキュリティの統合を重要な必須要件と位置づけ、当該製品の機能や市場投入スピードよりも優先して取り組むよう奨励している。エンジニアリングチームはいずれ、セキュリティが真に設計に組み込まれ、その維持も従来と比べ容易になる、新たな定常状態を確立できるようになるだろう。

こうした観点から、EU（欧州連合）はサイバーレジリエンス法において製品セキュリティの重要性を強化しており、ソフトウェアメーカーが脆弱な製品を流通させないように、製品のライフサイクルを通じたセキュリティを実装させるのが望ましいと強調している。

[1] 本書作成機関は、「Safety［安全・安全性］」という用語は、それが使用されている状況・背景次第で複数の意味を有していることを認識している。本ガイドにおいては、「Safety［安全・安全性］」とは、悪意あるサイバー活動から顧客を保護するためにテクノロジーセキュリティの水準を向上させることを指すこととする。

テクノロジーとその関連製品が顧客にとってより安全である未来をもたらすために、本書作成に携わった各機関は、ソフトウェアメーカーに対して設計・開発プログラムを見直し、セキュアバイデザインかつセキュアバイデフォルトである製品のみの出荷を許可するよう強く求めている。セキュアバイデザインである製品においては、顧客のセキュリティがただの技術上の機能のひとつではなく、主要なビジネス目標として、開発のはるか以前から概念化されている。セキュアバイデザインである製品は、開発開始前からそこを目標として始まるのである。既存製品でも、複数のイテレーションを経てセキュアバイデザイン状態へと進化することができる。セキュアバイデフォルトである製品とは、設定変更をまったく、あるいはほとんど要することなく購入後すぐに利用することができ、追加料金なしでセキュリティ機能が備わっているものである。これら 2 つの理念が共に実践されることにより、セキュアな状態を維持する負担の多くがソフトウェアメーカー側に移され、顧客が設定ミスやスピード感に欠けるパッチ適用、あるいはその他のよく見られる問題に由来するセキュリティインシデントの被害を受ける可能性が低減される。

米国サイバーセキュリティ・インフラストラクチャーセキュリティ庁（CISA）および国家安全保障局（NSA）、連邦捜査局（FBI）、そして下記の各国パートナー機関[2]は、本ガイド中の推奨事項を、ソフトウェアメーカーが自らの製品の安全を徹底するためのロードマップとして提示している。

- 豪州サイバーセキュリティセンター（ACSC）
- カナダ・サイバーセキュリティ・センター（CCCS）
- 英国国家サイバーセキュリティセンター（NCSC-UK）
- ドイツ連邦情報セキュリティ庁（BSI）
- オランダ国家サイバーセキュリティセンター（NCSC-NL）
- ノルウェー国家サイバーセキュリティセンター（NCSC-NO）
- ニュージーランド・コンピュータ緊急対応チーム（CERT-NZ）およびニュージーランド国家サイバーセキュリティセンター（NCSC-NZ）
- 韓国インターネット振興院（KISA）
- イスラエル国家サイバー総局（INCD）
- 日本内閣サイバーセキュリティセンター（NISC）およびJPCERTコーディネーションセンター（JPCERT/CC）
- OAS/CICTE政府ネットワーク 米州サイバーインシデント対応チーム（CSIRT）
- シンガポール・サイバーセキュリティ庁（CSA）
- チェコ共和国 国家サイバー情報セキュリティ庁（NÚKIB）

本書作成に携わった各機関は、多くの民間パートナー企業が設計由来の安全性（セキュリティバイデザイン）とデフォルトでの安全性（セキュリティバイデフォルト）の推進に貢献していることを認識している。本ガイドラインは、テクノロジーが設計上およびデフォルトで安全でセキュアかつ強靭である未来を実現するために必要となる優先取組事項や投資、意思決定についての国際的な対話を促すことを意図としている。この目的を果たすべく、本書作成機関は関係者・当事者からのこのガイドラインに対する意見を求めており、共通目的の達成に向けて我々のガイドラインをさらに洗練し、具体化し、前進させるための一連のヒアリングセッションを実施する予定である。

製品の安全性の重要性については、CISAの論文「The Cost of Unsafe Technology and What We Can Do About It［安全でない製品がもたらすコストと、その対策としてなし得ること］」を参照されたい。

[2] 以下、「本書作成機関」とする。

改訂にあたって

本報告書の初版は、ソフトウェア業界内で多くの会話を生み出すこととなった。組織や個人が侵入等の被害に晒されているというニュースを日々目にする現状は、ソフトウェア製品における慢性的かつ体系的な問題に対処する方法について、より多くの会話が持たれることの必要性を浮き彫りにしている。

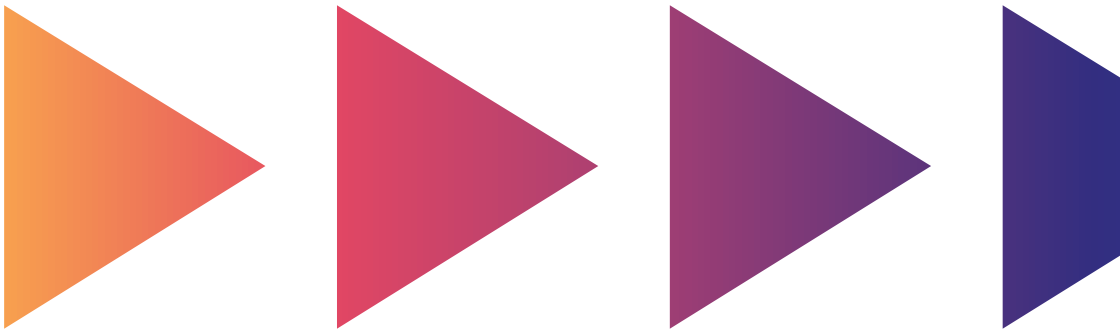
2023年4月の初版リリース後、本書作成機関（以下、これら各機関を指して「我々」という文言も使用する）のもとには、数百もの個人や企業、業界団体から思慮深いフィードバックが届けられた。フィードバックの中で最も多かった要望は、3つの原則についての詳細説明を提供することであり、これは同原則がソフトウェアメーカーとその顧客の両者にあてはまるためであった。本文書では報告書初版の内容を発展させ、ソフトウェアメーカーと顧客の規模、顧客の成熟度、そして原則の適用範囲など、初版では触れられていなかったテーマについても触れている。

ソフトウェアはあらゆるところに存在しており、ソフトウェアシステムやソフトウェア製品の開発、顧客におけるソフトウェアの導入とメンテナンス、そして他のシステムとの統合のすべてを単一の報告書で十分に網羅することは、不可能である。以下のガイダンスのうち、特定の環境に明確にマッピングされていないものについては、本文書で説明された慣行がどのようにして特定のセキュリティ上の改善へとつながったのか、ぜひコミュニティからの声をお聞かせいただきたいところである。

また、本報告書は、人工知能（AI）ソフトウェアシステムやAIモデルの作成者にも適用される。AIシステムやAIモデルは従来のソフトウェアとは異なるであろうとはいえ、根本的なセキュリティ上の慣行はAIシステム／AIモデルにも適用されるものである。一部のセキュアバイデザイン慣行については、AI特有の考慮されるべき点に対処すべく修正が必要となる可能性があるが、3つの包括的なセキュアバイデザイン原則は、すべてのAIシステムに適用される。

我々は、ソフトウェア開発ライフサイクル（SDLC）をセキュアバイデザイン原則に沿ったものとなるよう変貌させていく作業は単純なものではなく、時間も要するであろうことを認識している。加えて、小規模のソフトウェアメーカーは、こうした推奨事項の多くを実施することが困難となることも考えられる。我々は、ソフトウェア業界が製品を安全にするツールと手順を広く入手できるようにする必要があると考えている。我々は、ソフトウェアセキュリティの改善に目を向ける人々や組織が多くなるにつれてイノベーションの余地が生まれ、大規模・小規模ソフトウェアメーカー間にあるギャップが縮まり、すべての顧客の利益につながるだろうと信じている。

セキュアバイデザイン報告書初版からの今回の改訂版は、相互につながりを持ち、テクノロジーエコシステムを支えている多くのステークホルダー・コミュニティとのパートナーシップを構築するという、我々のコミットメントの一部を成している。この改訂版は、エコシステムの様々な箇所にいる方々から受けたフィードバックの結果であり、我々は今後も多様な視点からの意見に耳を傾け、学ぶことを続けていく。今後も多くの課題が待ち受けてはいるが、我々はセキュアバイデザインの理念を既に採用し、多くの場合は成功に至っている人々や組織について学ぶ中で、驚くほどに将来を楽観視している。



この文書の使い方

我々は、ソフトウェアメーカーがこの文書内に挙げられている原則を支持・厳守するよう強く求めている。ソフトウェアメーカーは以下に挙げられた手順に沿って、自らが実施した行動を公開文書化することにより、原則へのコミットメントを示すことができる。我々はソフトウェアメーカーに、この原則の精神に応じた手法を見つけ出し、懐疑的な現行顧客や潜在顧客に対しても説得力のあるかたちでセキュアバイデザインの理念を体現していると説明・主張できるアーティファクトをつくり出すよう奨励する。

本文書は、メーカーが取るべき行動の指針となることに加え、顧客側も活用することができる。ソフトウェアを購入する企業は、この文書に挙げられている原則を支持・厳守している事例から着想を得ながら、ベンダーに厳しい質問を投げかけるべきである。そうすることにより、顧客は製品がよりセキュアバイデザインとなる方向へと市場が移行するのを後押しすることができる。顧客がベンダーにたずねることができる質問の例は、「CISA's Guidance for K-12 Technology Acquisitions［K-12テクノロジー購入についてのCISAガイダンス］」に示されている。

我々は、エンタープライズ顧客が調達プロセスやベンダーのデューデリジェンス評価、エンタープライズ・リスク受容にかかる意思決定、そしてベンダー評価時のその他の手続きに、これらの慣行を組み込むことを奨励する。また、顧客は各ベンダーがセキュアバイデザインに関連して実施している行動を文書化して公開するよう、ベンダーに求めるべきである。顧客が集団としてこうした行動を取ることにより、安全性に対する需要が強く発信され、ソフトウェアメーカーによる安全性向上への取り組みの推進・実現へとつながる。つまり我々は、セキュアバイデザインの理念をソフトウェアメーカーに浸透させようという試みと並行して、メーカーの顧客とも「セキュアバイデマンド（需要由来の安全性）」の文化をつくりださなければならないのである。

セキュアバイデザイン

「セキュアバイデザイン」とは、テクノロジー製品が、悪意のあるアクターによる端末やデータ、ネットワークに接続されたインフラへの不正アクセス取得から適切に保護されるかたちで構築されていることを意味する。ソフトウェアメーカーはリスク評価を行い、重要システムに対してよく見られるサイバー脅威を特定・識別したうえで、サイバー脅威を取り巻く刻々と変化する環境を考慮した保護策を製品設計に織り込むべきである。

また、悪意のあるアクターによるシステムへの侵入やセンシティブデータへの不正アクセスを防ぐために、安全な情報技術（IT）開発慣行と、「縦深防御（Defense-in-Depth）」とも呼ばれる多層防御の仕組みが推奨されている。加えて、本書作成機関はソフトウェアメーカーに対し、システムに対するすべての潜在的脅威に対処し、各システムの導入プロセスにも配慮するよう、製品開発段階において自らにカスタマイズされた脅威モデルを使用することを推奨する。

本書作成機関はメーカーに、自らの製品やプラットフォームの安全について包括的なアプローチで取り組むことを強く求める。セキュアバイデザインに沿った開発では、製品の設計・開発プロセスの各層において、後付けすることができない専任リソースの戦略的な投資がソフトウェアメーカーに求められる。こうした開発においては、セキュリティをただの技術上の機能のひとつではなくビジネス目標として位置付けるという、メーカーの経営陣トップによる強いリーダーシップが求められる。このような経営陣と技術チームとの連携は、設計・開発の準備段階から始まり、顧客におけるソフトウェアの導入や保守にまで及ぶものである。ソフトウェアメーカーには、「顧客の目には見えない」もの（例：広範に及ぶ脆弱性を除去するプログラミング言語への移行など）を含む、厳しいトレードオフや投資を行うことが推奨される。メーカーは、顧客にとって魅力に見えるものの攻撃対象領域を拡大させてしまうような機能ではなく、顧客を保護する機能、仕組、そしてツールの実装を優先するべきである。

悪意のあるアクターによる技術の脆弱性の悪用という継続的な脅威をただひとつのソリューションで終結させることは不可能であり、今後も「セキュアバイデザイン」である製品からも脆弱性が発見されつづけることは間違いないが、脆弱性の多くは比較的少数の根本原因に由来するものである。ソフトウェアメーカーは、既存の製品ポートフォリオをセキュアバイデザインの慣行に沿ったものとするためのロードマップを文書として作成し、その慣行から外れるのは例外的な状況でのみとするよう徹底すべきである。

本書作成機関は、顧客にもたらされるセキュリティ上の結果をメーカーが自らの責任で管理し、この水準で顧客のセキュリティを確保することが、開発コストの増加につながり得ることを認識している。しかし、革新的なテクノロジー製品の開発と既存製品の維持・保守を行いつつセキュアバイデザイン慣行に投資することにより、顧客の安全態勢の大幅な改善と侵入等の被害を受けるリスクの大幅な低減が可能となる。セキュアバイデザインの原則は、顧客の安全態勢と開発業者のブランド評価を高めるだけではなく、その慣行の実践により、長期的にはメーカー側の保守およびパッチ適用コストの低減も実現されるのである。

本文書中のこの後に記している「ソフトウェアメーカーへの推奨事項」の項では、メーカーが検討すべき製品開発関連の慣行および方針の項目がまとめられている。

セキュアバイデフォルト

「セキュアバイデフォルト」とは、製品が追加料金なしで購入後すぐにでも、よく見られる悪用・侵入等の手法に対して強靭な対応力を持つ状態にあることを意味する。こうした製品は、エンドユーザが安全を確保するために追加措置を講じる必要なく、最もよく見られる脅威や脆弱性に対する保護をもたらすものである。セキュアバイデフォルトの製品は、安全な初期（デフォルト）設定から外れた状態になったときに、追加の補助制御策を導入しなければ侵入等の被害に遭う可能性が高まることを顧客に強く自覚させるよう設計されている。セキュアバイデフォルトは、セキュアバイデザインの一形態である。

- 安全な設定は、初期設定（デフォルト）のベースラインであるべきである。セキュアバイデフォルトの製品においては、悪意のあるアクターからエンタープライズを保護するために必要な最も重要となるセキュリティの制御が自動で有効化され、追加料金なしでセキュリティ制御の使用やさらなる詳細設定を行う能力が提供される。
- セキュリティ設定の複雑さは、顧客側の問題とすべきではない。顧客組織のITスタッフは、セキュリティ面および運用面の責務で過負荷の状態に陥っていることが多く、その結果、堅牢なサイバーセキュリティ態勢に必要とされるセキュリティ上の意味合いとリスク緩和策の理解・導入に割ける時間が限定されてしまっている。ソフトウェアメーカーは、安全な製品設定を最適化する— すなわち「デフォルトパス」の安全を確保することで顧客を支援し、自社製品が必ず「セキュアバイデフォルト」基準に従い安全に作成・配布・利用されるよう徹底することができる。

「セキュアバイデフォルト」である製品のメーカーは、追加のセキュリティ設定を実装するために追加料金を請求しない。こうしたメーカーはそのような追加課金を行うのではなく、すべての新車にシートベルトが付いているように、セキュリティ設定をベースラインナップの製品に標準で搭載している。

セキュリティは贅沢なオプションではなく、顧客が交渉や追加料金を要することなく享受できる権利として考えられるべきである。

ソフトウェアメーカーへの推奨事項

この共同作成ガイドはソフトウェアメーカーに対し、ITセキュリティを導入・徹底するためのロードマップを文書として作成する際の推奨事項を提供している。本書作成機関はソフトウェアメーカーに対し、セキュアバイデザインおよびセキュアバイデフォルトの原則を通して顧客にもたらされるセキュリティ上の結果を自らの責任で管理するために、下記の各項で概説する戦略を実行するよう推奨する。

ソフトウェア製品のセキュリティ原則

ソフトウェアメーカーには、ソフトウェアセキュリティを優先することを戦略的重点のひとつとすることが推奨されている。本書作成機関は、ソフトウェアメーカーが製品を開発・設定・出荷する前の設計プロセスにソフトウェアセキュリティを組み込むうえでの指針となるよう、以下の3つの主要原則を策定した。

1

顧客にもたらされるセキュリティの上の結果を自らの責任で管理し、それに合わせて製品を発展・進化させる。セキュリティ上の負担は、顧客のみが負うべきではない。

2

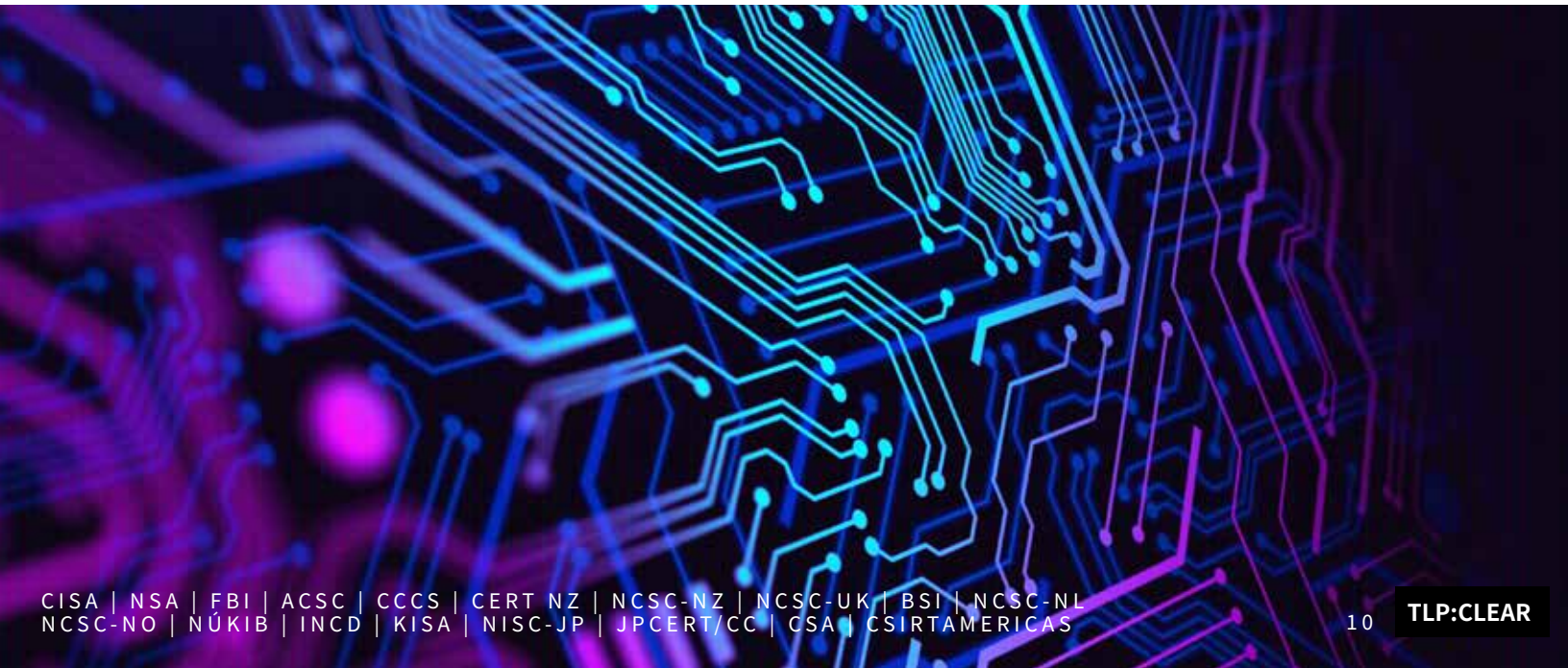
徹底的な透明性と説明責任を確保・推進する。

ソフトウェアメーカーは、安全でセキュアな製品を提供すること、そしてそうした製品を提供する能力に基づいてメーカーコミュニティにおける他業者との差別化を図ることに、誇りを持つべきである。ここにはたとえば、強力な認証メカニズムのデフォルト採用の普及度合いなど、顧客における導入事例を通して学ばれた情報の共有なども含まれ得る。また、脆弱性アドバイザリとそれに関連する共通脆弱性識別子（CVE）の登録が完全かつ正確であるよう徹底するための強いコミットメントも、ここに含まれる。ただし、CVE件数は健全なコード分析とテストコミュニティが存在する証でもあり、CVE件数をネガティブな指標としてみなしてしまう誘惑には注意が必要である。

3

上記目標の達成に向けた組織体制と経営リーダーシップを構築する。

製品の安全には技術的な専門知識が不可欠であるが、組織改革の実行に際しての主たる意思決定者は経営陣上層部である。経営陣は、セキュリティを製品開発の重要な要素として組織全体で、そして顧客とのパートナーシップのもとで、優先させなければならない。



これらの3つの原則の実現に向けて、ソフトウェアメーカーは自らの開発プロセスを進化させるために、以下のようないくつかの実務的な戦術手法を検討すべきである。

経営陣トップとの定例会議を開催して、組織内におけるセキュアバイデザインおよびセキュアバイデフォルトの重要性を推進する。上記原則に従った製品を開発したチームにはその見返りとして報酬が与えられるよう、社内の方針および手続が策定されるべきである。こうした報酬としては、優れたソフトウェアセキュリティ慣行の導入に対する賞の授与や、キャリア昇格や昇進要件におけるインセンティブなどが考えられる。

事業の成功におけるソフトウェアセキュリティの重要性を中心に据えて、業務を遂行する。たとえば、ソフトウェアセキュリティ水準とメーカーの説明責任を直接結びつけるようなビジネス慣行およびIT慣行の実施・順守を担う「ソフトウェアセキュリティ・リーダー」や「ソフトウェアセキュリティ・チーム」の任命を検討する。ソフトウェアメーカーは自らの製品について、強固で独立した製品セキュリティ評価・査定プログラムを有しているよう徹底すべきである。

リソースの配置・割当および開発に際してはカスタマイズされた脅威モデルを使用し、最も重要かつ影響の大きな機能に優先的に取り組む。脅威モデルは製品固有のユースケースを検討し、開発チームによる製品の強化を可能にしてくれるものである。最後に、経営陣トップは各チームに対して、セキュアな製品の提供が製品の卓越性や品質の中心要素となるよう、責任を負わせるべきである。

本指針の2023年10月改訂の一環として、上記の3原則は以下に挙げる詳細説明や実証、エビデンスを通して発展を遂げている。

原則1：顧客にもたらされるセキュリティ上の結果を自らの責任で管理する

詳細説明

最新のベストプラクティスは、ソフトウェアメーカーに対して**アプリケーションのハードニング（堅牢化）、アプリケーションの機能、**そしてアプリケーションの**初期（デフォルト）設定**を含む、製品セキュリティに関連した取り組みに投資するよう指示している。

ソフトウェアメーカーは、アプリケーションに侵入等の被害をもたらそうとする悪意のあるアクターのコストを高くするようなプロセスおよびテクノロジーを使用して、**アプリケーションの堅牢化**を行わなければならない。アプリケーション堅牢化のプロトコルと手順は、製品が知識を備えた悪意のあるアクターからの攻撃に抵抗する際の手助けとなる。ハードニング（堅牢化）、製品セキュリティ、レジリエンス（強靭性）といった用語はすべて、製品品質と密接に関連している。これはつまり、セキュリティとは「後付けされる」のではなく、「元から練り込まれて（組み込まれて）」いなければならないという考え方である。[1] セキュリティを組み込むことにより、ソフトウェアメーカーは顧客のセキュリティだけでなく、自らの製品品質も向上させることができる。こうした戦術の例としては、ユーザによる入力データが検証・サニタイジングされ、コードに直接入力されないよう徹底する（すなわち代替としてパラメータ化されたクエリを使用する）ことや、メモリセーフなプログラミング言語を使用すること、厳格なソフトウェア開発ライフサイクル（SDLC）管理、そしてハードウェアで補助されている暗号鍵管理を使用すること、などが挙げられる。

アプリケーションは、サイバーセキュリティに関する**アプリケーション機能**をサポートしていなければならない。「能力／性能」とも呼ばれることがあるこれらの機能は、顧客のセキュリティ態勢の維持ないし向上

に役立つかたちで製品またはサービスの機能性を拡張するものである。セキュリティ関連の機能の例としては、すべてのネットワーク接続に対するトランスポート・レイヤー・セキュリティ（TLS）のサポート、シングルサインオン（SSO）のサポート、多要素認証（MFA）のサポート、セキュリティイベントの監査ログの記録、役割ベースのアクセス制御（RBAC）、属性ベースのアクセス制御（ABAC）などが挙げられる。

こうした製品機能の中には設定変更可能なものもあり、顧客がより容易に既存の環境やワークフローへの製品の統合を行えるようにしている。それらの設定が意味するのは、アプリケーションは顧客が設定するまで**デフォルト設定**が施されていなければならないということである。これらのデフォルト設定は、顧客が所有する大量のテクノロジー製品をより安全にするために費やすリソースを削減できるように、「購入後すぐに」安全に設定されていなければならない。

アプリケーションの堅牢化、アプリケーションセキュリティ機能、アプリケーションのデフォルト設定というこれらの各要素は、アプリケーションのセキュリティとその結果としてもたらされる顧客のセキュリティ態勢において役割を果たすものである。ソフトウェアメーカーは、これら各要素について、そして各要素間の関連性について考慮するべきである。また、メーカーはこれらの要素を製品に組み込むための投資だけでなく、より広くその先まで考えるべきである。メーカーはさらに一歩先に進み、これらの要素が顧客の現実世界におけるセキュリティ態勢を良くも悪くもどのように変化させるのかについて考慮すべきである。

メーカーはその努力や投資によってのみ自己評価をするのではなく、顧客にもたらされるセキュリティ上の結果を自らの責任で管理すべきである。その責任は、侵入等の被害のリスクを低く抑えることができる可能性が最も高い上流側、つまりメーカー側に置かれるべきである。

残念ながら、現状はそのようなかたちにはなっていない。包括的な**アプリケーション堅牢化**に投資するのではなく、セキュリティの責任を顧客に押し付けているソフトウェアメーカーがあまりにも多い。たとえば、メーカーがある脆弱性にパッチを適用しても、それはその欠陥の根本原因への対応ではなく症状への対処（対症療法）であるために、同様の脆弱性が明らかになることも珍しくない。当該製品は、同じクラス属性を持つ脆弱性に対してコードベースの様々な箇所に異なる緩和策を実装することになる可能性がある。その典型として、メーカーがある入力サニタイ ズの脆弱性を修正した後、研究者や攻撃者が、改善された入力サニタイズの恩恵から外れているコードパスを発見した事例があった。このとき、メーカーはコードベースを統一してアプリケーション全体において同じクラスの脆弱性を完全に除去するのではなく、一度にひとつずつ修正を適用するという対応を繰り返し行った。

アプリケーションの機能は、顧客にメリットとリスクの両方をもたらし得る。一般的に、多くの外部システムやバージョンとの結合（統合）ポイントを提供する機能は、製品の価値を大きく高めることができる。しかし、ネットワークプロトコルのように廃棄プランがない機能へのサポートは、顧客がその機能を使い続けることが何を意味するのかを十分に理解していない場合に、その顧客に脆弱性をもたらしてしまう可能性がある。たとえば一部の製品では、元々1990年代や2000年代にリリースされ、現在では安全でないことが知られているネットワークプロトコルが未だに使われている。顧客によるアップグレードの実行や最新セキュリティ対策の導入を遅らせ得る要因は、数多くある。顧客が使用している製品が、組織内の他のネットワークとの統合はできても最新のセキュリティ対策を欠いていて、IT部門による近代化・最新化の妨げとなっている場合もある。それでもなお、ソフトウェアメーカーはこうしたパターンをプランニング・プロセスの中に織り込んでおき、最新状態を維持するよう顧客に奨励することができる。

アプリケーションの初期設定は、顧客にとって潜在リスクとなり得るもうひとつの領域である。メーカーは、特定のデフォルト設定を選択し、顧客が望むアプリケーション機能を使いやすくすることが少なくない。こうした慣行のマイナス面は、デフォルト設定により有効にされている特定の機能やプロトコルを必要としない顧客の攻撃対象領域を拡大してしまう点である。加えて、セキュリティ制御はその多くがデフォルトで無効に設定されていたり、セキュリティ向上のためには顧客が時間をかけて設定変更を行わなければならない状態になっている。明示的な脅威モデリングは、どの機能をデフォルトで有効にしておくべきかや、セキュアバイデフォルトの状態にはどの設定が必要なのかを決定する際の判断要因として役立ち得る戦術手法である。別の戦術手法としては、管理者が機能をより発見しやすくするための方法を検討することが挙げられる。

デフォルト設定が一部あるいはすべての顧客にリスクを生じさせる可能性がある状態になっている製品を出荷しているメーカーもある。こうしたメーカーは、より安全なデフォルト設定を施す代わりに、顧客側がコストを負担して導入しなければならない**堅牢化（ハードニング）ガイド**を作成していることが少なくない。堅牢化ガイドは、その作成者が誰であるかに関わらず、いくつかの共通した問題を抱えている。まず、こうしたガイドの中にはその存在が見つけにくく、サポートも十分でないものがある。そうでなかったとしても、導入が複雑で拡張モジュールを書くためのソフトウェア開発が必要となる場合もある。加えて、ガイドの読み手が様々な設定によりどのように攻撃対象領域が変化するのかを理解するために必要な、豊富なサイバーセキュリティ経験を有していることを前提としているようなものもある。攻撃側がどのように行動するのかを十分に理解していない実務者は、堅牢化ガイドの指示を適切に導入できない可能性があり、そうした指示においてトレードオフが明確に示されていない場合は、とりわけその可能性が高くなる。さらに、すべての堅牢化ガイドが攻撃者の戦術手法や経済原理に精通したエンジニアにより作成されているわけではないため、仮に忠実に導入されても実効性がないガイドが作成されてしまうこともある。リソースが限られている環境にあることが少なくない中で、何百万という顧客が複数インスタンスのソフトウェアやシステムを堅牢化する責任を負わされているのである。堅牢化ガイドに頼るやり方では、単純に釣り合いが取れない。

アプリケーションの設定は、設定がメーカーによるデフォルトであろうと顧客によるものであろうと、脅威を取り巻く環境についてのメーカー側の現状理解に照らして継続的に評価されるべきである。アプリケーションは、そうした設定からもたらされる潜在的なリスクについての明確な指標とともに作成されるべきであり、そのような指標は公開されるべきである。最新の自動車ではシートベルトについてのインジケーターが備わっていて、シートベルトを締めずに運転しようとするとアラームが鳴ってそのインジケーターが表示されるように、ソフトウェアでもシステムのセキュリティ状態についてのインジケーターが表示されるべきなのである。管理者アカウントに多要素認証（MFA）を要求しないよう設定されているアプリケーションがあるのなら、そのアプリケーションは、MFAを設定しなければ自身とその組織全体が危険に晒されていることを管理者に定期的に認識させるべきである。加えて、今日では脆弱な暗号が実装されていると知られているような古いプロトコルをサポートするように設定されているアプリケーションがあるのなら、そのアプリケーションは、当該組織が危険に晒されていることを管理者に明確かつ定期的に伝え、その状況を解決するためのリソースも提供するべきである。我々はメーカーに対し、堅牢化ガイドを解釈するための時間や専門ノウハウおよび認識を確保するよう顧客側の管理者に求めることに頼るのではなく、製品に組み込まれた定期的なナッジメッセージを導入するよう、強く求める。セキュリティと利便性への配慮のバランスを実現するためのイノベーションの実現に向けては、明らかに改善の余地が存在している。

上記の各要素は、顧客が侵入等の被害のリスクを低減するために追加の**セキュリティ製品**についての調査や予算支出、購入、人員割当て、導入、モニタリングを行わなければならないという容認し難い状況を作り出している。一般に、中小規模の組織（SMO）がこうした選択肢を採用・実施することは不可能である。SMOでは専門ノウハウや予算、時間が不足しており、それが対応力と機能性に負担を掛けてセキュリティ対策の優先順位を下げてしまい、そのような組織が多くなることで、業界全体のリスクも高まってしまうこととなる。逆に、比較的少数のメーカーによるセキュリティ投資は、釣り合いが取れる。この問題を要約してよく使われるのは、ソフトウェア業界が必要としているのはセキュリティ製品の数を増やすことではなく、よりセキュアな製品である、というフレーズである。この改革は、ソフトウェアメーカー主導で進めることが望ましい。

> ***ソフトウェア業界が必要としているのは、セキュリティ製品の数を増やすことではなく、よりセキュアな製品である。この改革は、ソフトウェアメーカー主導で進めることが望ましい。***

今日、特定のセキュリティ機能を有効にしていなかったため、あるいは特定の堅牢化ガイダンスに従わなかったために顧客が侵入等の被害を受けたと説明するメーカーのコメントを目にすることがある。メーカーはそのようなコメントを出すのではなく、侵入等の被害が発生した後に特定のセキュリティ機能または特定の堅牢化ガイダンスがあればその被害を防げていたのかどうかを説明し、そのような機能やガイダンスを追加料金無しでデフォルト化することを検討するべきである。製品自体の設計・導入段階で十分に堅牢化されていなかったのであれば、メーカーは自社の製品ラインから同じクラスの脆弱性を排除するためにどのような取り組みを進めているのかを説明するべきである。

ソフトウェアメーカーには、製品がセキュリティを最優先にして設計・開発されていることを徹底する責任がある。そのためには、関連分野における自らの取り組みの**成果を客観的に測定する**べきである。我々はメーカーに対して、自らの社内努力のみに集中するのではなく、製品セキュリティに関する努力と設定の結果および実効性について、客観的に測定し定期的に報告すること、そして顧客の安全を測定可能なかたちで改善することに繋がるようなソフトウェア開発ライフサイクル（SDLC）上の変化を生み出すフィードバックと対応の循環的な仕組みを構築することを要望する。ここでの報告には、学術界やセキュリティ業界のリサーチ・コミュニティが、エコシステム全体規模の大枠での動向追跡を進捗測定に活用できるような、匿名データを含めるべきである。

この原則の実施を実証するために

ソフトウェアメーカーとオンラインサービス事業者は、この原則の実施成功を実証する方法を特定するべきである。メーカーは、第三者が調査できるようにアーティファクトのかたちで証拠を提供できるよう努めるべきである。メーカーが強固なセキュアバイデザイン・プログラムを実施していることを単一のアーティファクトで証明することは不可能だが、複数のアーティファクトを提示することで、そのメーカーのセキュアな製品開発へのコミットメントを証明する論拠を構築することができるだろう。このアプローチは、「語るより示せ」の精神に基づいている。

この原則を実践するために、ソフトウェアメーカーは以下のリストに挙げるような手順を検討するべきである：本書作成機関は、セキュアバイデザインに向けての取り組みに着手した時点で、これらの慣行を直ちに導入して対応するアーティファクトを作成できるメーカーなどほぼ皆無であろうことを認識している。さらに、ソフトウェアメーカーは、顧客が製品を実際にどのように実装しているかに基づいて、セキュリティ上最大限の効果を実現するために、下記のリストに優先順位をつけることが必要となるだろう。

セキュアバイデフォルトな慣行

1. **デフォルト・パスワードを撤廃する。**初期（デフォルト）パスワードは、これまで同様に、毎年多くの攻撃の原因であると示唆されている。この慢性的な問題の根絶に責任をもって取り組むことは、攻撃者による容易なアクセス取得の防止につながる。同様に、メーカーはパスワードについて、最低限の長さや、過去に破られたことが把握されているパスワードの禁止など、どのようなパスワード慣行を導入すべきか検討するべきである。
2. **フィールドテストを実施する。**技術が継続的に進化して複雑化するにつれて、ソフトウェアメーカーが現場でのセキュリティに焦点を当てたユーザテストを実施し、現場における自社製品のセキュリティ態勢を把握することの重要性は、ますます高まっている。ソフトウェアメーカーは、ユーザ調査から得た情報に基づいてソフトウェア開発上の要件を把握するのと同様に、セキュリティに焦点を当てたユーザ調査を実施してセキュリティ面でのユーザ体験（UX）が不十分である点を把握するべきである。ソフトウェアメーカーは、顧客が実際の環境で自社製品をどのように導入・使用しているのかを観察することで、そのソフトウェアのセキュリティ機能および制御の使いやすさや有効性についての貴重な知見を得ることができる。こうした知見は、改善すべき部分を特定して自社製品を洗練し、顧客がセキュリティ面で抱えるニーズに対してより的確に応えられるようにするため役立てられる。たとえば、フィールドテストの結果からUXフローやデフォルト設定項目、警告の詳細、モニタリング等の変更が示唆されることが考えられる。また、フィールドテストを行うことで、過去に当該ソフトウェアの設計に施された改善により、セキュリティパッチ適用速度の減速や、設定エラーの削減、そして攻撃対象領域の最小化が実現された箇所が明らかになる場合もある。

メーカーは、以下の点を検討すべきである：

- 顧客は堅牢化ガイドを正しく実施しているか？
- 製品の既存セキュリティ機能は、現場で期待通りに機能しているか？
- こうした機能は実際の攻撃に対して抵抗力を発揮しているか？
- 侵入等の被害の可能性をより低減できるのは、どの機能か？

注：これらの側面についてより深い知見を得るために、ソフトウェアメーカーは顧客と協力してレッドチーム演習を実行し、製品がどのように攻撃に抵抗するのかを確認することを検討すると良いかもしれない。これらのフィールドテストは、顧客の物理的なサイトで行うこともできるし、仮想サイトや、プライバシーを保護したかたちでアプリケーションからのテレメトリ（遠隔測定）で行うこともできる。

3. **堅牢化ガイドのサイズを縮小する。**メーカーは、堅牢化ガイドを簡素化、場合によっては撤廃し、顧客が自社製品を導入する際に優先すべきセキュリティ対策に注力することで、顧客のセキュリティ態勢を改善することができる。メーカーは、セキュリティ対策を羅列して顧客を困惑・閉口させるのではなく、自社製品が影響を受けやすいことから優先順位が高いセキュリティリスクを特定し、それらのリスクを緩和する方法について明確かつ簡潔なガイダンスを提供するべきである。加えてメーカーは、顧客側の環境に容易に導入できるスクリプトなど、セキュリティ制御の導入プロセスを簡素化するツールや自動化機能を顧客に提供するべきである。こうしたツールは、もととなるベースライン状態に対して施された変更を追加的に検証し、明確に示すことができるのが望ましい。メーカーは、堅牢化ガイドの簡素化と、使いやすいツールおよび自動化機能を提供することで顧客の負担を軽減し、自社製品がセキュアな方法で導入されるよう徹底するための補助をすることができる。こうした取り組みにおける戦術手法のひとつとして考えられるのは、パレートの法則の実践を検討し、よくあるユースケース（80%）では手順数を減らし、そのうえで一般的ではないシナリオ（残りの20%）については状況に適したガイダンスとツールを提供するというものである。ソフトウェアメーカーはこのような方法で、単純なことは単純化しつつ、困難なことも実現可能な状態にできる。フィールドテストは、顧客が堅牢化ガイドを発見して理解し、実施するのにどの程度の時間を要するのかを測定するための強力なツールとなる。また、メーカーは管理者が堅牢化ガイドからタスクを実行することに頼るのではなく、製品からどのようなナッジングがあれば管理者にその製品内で行動を取るよう促せるのかを検討すべきである。

4. **安全ではないレガシー機能の使用を積極的に非推奨とする。**
明確なアップグレードパス（行程）を提供することで、後方互換性を持たせることよりもセキュリティを優先する。より安全な機能やプロトコルの採用について記載したブログを公開して、安全ではない機能については非推奨であると公表し、この公表については製品内の通知も選択肢として検討する。相当数の顧客が、最新のネットワークやIDその他の重要なセキュリティ機能を利用してシステムを最新状態に維持することはしないと示されている。顧客が、アップグレードをすることで既存の機能が壊れてしまうのではないかと恐れている場合もある。アップグレードを可能な限りシームレスに行えるようにすることで、顧客がより頻繁かつ迅速にアップグレードを実行し、セキュリティ修正を行う見込みも高くなる。ソフトウェアメーカーは、顧客側のリスクを軽減するようなアップグレード行程への顧客のナッジングを精力的に行うべきである。

5. **目を引くアラートを導入する。**
メーカーは、シートベルト未着用の際に繰り返し音を発する自動車のシートベルト警報と同様の、ユーザや管理者が真に危険な状態にあるときに適時繰り返しアラートを発し、環境内で非推奨のプロトコルが使用されていることを管理者に警告してアップグレード行程を提案する仕組みを導入すべきである。ユーザや管理者、あるいはアプリケーション設定が安全ではない状態にあるときに、適時的に繰り返しアラートを発する仕組みを導入する。安全でないモードにあるときは、その事実を管理者に頻繁かつ明確に伝える。追加機能として、スーパー管理者がログインするたびに自分のアカウントが多要素認証（MFA）になっていないことを承認するよう求めたり、それだけでなくMFAを有効にするまで特定の主要機能を無効にしたりすることも考えられる。アラート疲れを起こさないようにしつつこれらの目標を達成できるようにするためには、イノベーションの余地がある。

6. **セキュアな設定テンプレートを作成する。**
こうしたテンプレートは、組織のリスク許容度に応じて特定の設定項目を安全な設定にプリセットすることができる。セキュリティテンプレートを低・中・高の各レベルで設定しておくのではあまりに単純すぎるかもしれないが、この3レベルの例では、組織のリスクを管理するためにいくつの設定項目を更新すればいいのかが示されている。テンプレートは、メーカーが特定したリスクについての堅牢化ガイドでサポートすることができる。

セキュアな製品開発慣行

1. **安全なSDLC枠組みへの適合性を文書化する。** 安全なSDLC（ソフトウェア開発ライフサイクル）枠組みでは人、プロセス、技術にわたる目的と事例が提供されている。安全なSDLC枠組みにおける制御のうちどのようなものが導入されたのかについての詳細説明の公表を検討し、使用された代替制御の説明も示すこと。米国内であれば、NIST のセキュアソフトウェア開発フレームワーク（SSDF）の使用を検討すること。SSDFはチェックリストではないが、「安全なソフトウェア開発のための基本的かつ健全な一連の慣行を記述している」。

2. **サイバーセキュリティ・パフォーマンス目標（CPG）または同等の基準への適合性を文書化する。** 組織がNISTのSSDF基準に適合していると主張する場合、それは当該組織が自らのSDLC が十分に理解されたベストプラクティスを考慮したうえで策定されていると主張していることを意味する。しかし、強固なSDLCを有しているだけでは十分ではない。まだ開発段階にある製品のセキュリティ特性に手を加えて操作しようとする悪意のあるアクターから、自らの企業と開発環境を保護することも必要となる。これは理論上のものと分類される攻撃ではなく、既に現実に実行され、顧客にとどまらず国家安全保障にまで有害な影響を及ぼしている攻撃である。組織はCISA CPGおよびNISTのサイバーセキュリティ・フレームワーク（CSF）、またはその他のサイバーセキュリティ・プログラムの枠組みに対する自らの適合性についての詳細の公表を検討すべきである。

3. **脆弱性管理。** 一部のメーカーの脆弱性管理プログラムでは、社内または社外で発見された脆弱性へのパッチ適用に集中し、それ以外のことがほとんど行われていない。より成熟したプログラムでは、脆弱性とその根本原因に関する広範囲に及ぶデータに基づいた分析が取り入れられており、同じクラスの脆弱性全体を体系的に排除するための措置が取られている[3]。このようなケースでは、品質計画や品質管理、品質改善、品質測定に基づいた正式なプログラムが導入されている。成熟したプログラムにおいては、欠陥管理が単なるセキュリティ問題ではなく、ビジネス上の問題とみなされる。これらのプログラムは、見方によっては他業界における品質および安全関連のプログラムと同様のものである。

4. **オープンソースソフトウェアは責任を持って利用する。** オープンソースのソフトウェアが利用される場合には、オープンソースパッケージの綿密な調査や、プログラム依存関係へのコード貢献の奨励・促進、そして重要コンポーネントの開発・保守の支援を通して、責任を持って利用すること。参考までに、日本の経済産業省（METI）は「Collection of Use Case Examples Regarding Management Methods for Utilizing OSS and Ensuring Its Security」［オープンソースソフトウェアの利活用及びそのセキュリティ確保に向けた管理手法に関する事例集］を公開している。

5. **開発者のための安全なデフォルトを提供する。** ソフトウェア開発期間中は開発者向けに安全なビルディング・ブロックを提供することで、デフォルトルートが安全なものとなるようにする。たとえば、SQLインジェクションの脆弱性が蔓延して現実世界で被害を生んでいることを踏まえ、開発者がしっかりとメンテナンスされたライブラリを使用して同クラスの脆弱性を防ぐよう徹底する。「Paved Roads［舗装道路］」や「Well-Lit-Paths［明るく照らされた道］」とも呼ばれるこの慣行では、スピードとセキュリティの双方が確保され、人的エラーも削減される。

6. **セキュリティを理解したソフトウェア開発者を育成する。** 社内・組織内のソフトウェア開発者に安全なコーディングについてのベストプラクティスのトレーニングを受けさせ、開発者がセキュリティを理解しているよう徹底する。加えて、候補者のセキュリティ知識を評価するように採用慣行を改訂したり、大学やコミュニティカレッジ（短大・専門学校）、ブートキャンプその他の教育機関と協力してコンピュータ・サイエンスやソフトウェア開発のカリキュラムにセキュリティ項目を組み込むことで、より広範な業界労働者の変革を後押しする。

[3] NIST SSDF, PO 1.2, 事例2:「その組織のソフトウェアについてのセキュリティ要件を定めた方針を定め、SDLCの各主要ポイントで要件が順守されているかどうかを検証する（例：一連のソフトウェア不具合というクラス属性を持つものを各ゲートで検証、リリース済みソフトウェア内で発見された脆弱性へのレスポンス等）」

7. **セキュリティインシデント・イベント管理（SIEM）と、セキュリティのオーケストレーションと自動化によるレスポンス（SOAR: Security Orchestration, Automation, and Response）の統合テストを行う。**
フィールドテストの実施に加えて、広く利用されているSIEMおよびSOAR業者との協力のもと、選定した顧客と共に、インシデント対応チームがセキュリティインシデントが疑われる事案や実際のセキュリティインシデントの調査においてどのようにログを使用するのかを理解する。ソフトウェア開発者はインシデント対応の経験を有していることが稀であるため、実際の対応担当者が期待しているほどの有用性を持たないログエントリを作成する可能性がある。SIEMとSOAR両方の技術に触れ、実際のインシデント対応専門家と共に作業することで、ソフトウェア開発チームは正しく完全に経緯を伝えるログを作成することができるようになり、インシデント発生時の時間短縮や不確実性／曖昧さの削減につながる。

8. **ゼロトラストアークテクチャ（ZTA）との整合を図る。**製品導入ガイドが、たとえばNISTのZTAモデルやCISA Zero Trust Maturity Modelと整合されているようにする。顧客が自らの環境にこれらの原則を取り入れるよう奨励する。

セキュリティに能動的なビジネス慣行

1. **追加料金なしでのログ記録機能の提供。**クラウドサービスは、セキュリティ関連のログを追加料金なしで生成・保存することにコミットするべきである。同様に、オンプレミス製品もセキュリティ関連のログを追加料金なしで生成するべきである。さらに、多くの顧客が実際にインシデントが発生するまでセキュリティイベント・ログの価値を理解せずにいることも珍しくないことから、こうした製品はデフォルトでセキュリティイベントのログ記録を取るべきである。この戦術手法では、サイバーセキュリティの現状認識をもたらすためにどのようなセキュリティイベントをログ記録するべきなのかや、顧客がどのようにログ記録を設定する可能性があるのか、ログはどのくらいの期間保持されるのか、ログの完全性とストレージを保護する期間、そしてログ分析はどのように行われ得るのか、といった項目についての徹底的な見直しを要する可能性がある。場合によってはこの見直しが、見直された項目を実行可能な状態にするために、メーカー側にとっても対応可能なコストでアプリケーションログの管理アーキテクチャをリファクタリングする必要性が示されることもある。インシデント対応(IR)の専門家と作業をすることにより、ログ記録が現場の調査担当者にとって有用となる可能性を高めることができる。SIEMの項も参照すること。

2. **隠れたコストを排除する。**セキュリティやプライバシー機能、統合などに決して追加料金を課さないというコミットメントを公表する。たとえば、広義でのIDアクセス管理（IAM）の中にはSingle Sign On（SSO）と呼ばれるサービスがある。一部のメーカーでは、自社システムのSSOサービス（IDプロバイダとも呼ばれる）への接続に追加料金を課している。この「SSO税」により、優れたIDアクセス管理は多くの中小規模組織（SMO）にとって高額で手が届かないものとなり、そうした組織における強固なセキュリティ態勢の実現の妨げとなっている。また、ユーザに対してMFAを有効にすることに追加料金を課しているサービスもある。**セキュリティは贅沢品として価格設定されるのではなく、顧客の権利とみなされるべきである。**一部のメーカーは、これらの機能は顧客からの需要もほとんど無く、維持にもコストがかかると主張してきた。こうした主張は、クレームや価格交渉の電話をかけてくる顧客などほとんどいないという事実や、すべての顧客がこれらの機能のメリットを理解しているわけではないこと、加えてそもそもすべての機能にはなんらかの維持費がかかるという点を無視したものである。一方で、メーカーが可用性やデータの完全性に追加料金を課すケースはあまり多く見かけない。こうした重要な機能特性をサポートするコストは、交通事故の際に命を守るシートベルトや衝撃吸収型のステアリングカラム、エアバッグなどを搭載するためのコストと同様に、すべての顧客が支払う料金に織り込まれている。

3. **オープンスタンダードを積極的に利用する。**オープンスタンダードを導入し、とりわけ共通ネットワークおよびIDプロトコルにはその導入を進める。オープンスタンダードが利用できる場合には、独自仕様のプロトコルを避ける。

4. **アップグレードツールを提供する。**セキュアネットワーク接続などのより安全性の高い機能の実装を含め、最新バージョンの製品の採用には多くの顧客が積極的でない。ソフトウェアメーカーは、不確実性やリスクの軽減につながるツールを提供することにより、顧客による新たなアップグレードの採用数を向上させることができる。顧客にアップグレードを動機づける手段として、テスト環境でアップグレードやパッチを検証できる無料ライセンスを提供すること。

原則2：徹底的な透明性と説明責任を確保・推進する

詳細説明

ソフトウェアメーカーは、安全でセキュアな製品を提供すること、そしてそうした製品を提供する能力に基づいてメーカーコミュニティにおける他業者との差別化を図ることに、誇りを持つべきである。

透明性に関する一般的な懸念に目を向けてみよう。実務者が「徹底的な透明性」について論じると、「攻撃者のためのロードマップ」を提供しているという懸念が出てそこから話が進まなくなる傾向がある。しかし、エビデンスが圧倒的に示しているのは、攻撃者はそのようなロードマップがなくてもまったく問題なく活動しているという事実であり、上記のような懸念よりも、直接顧客や間接顧客、サプライチェーン、そしてソフトウェア業界全体に恩恵をもたらす透明性のほうが優先されるべきである。

透明性は、業界による慣例― つまり、「良い状態」とはどのような状態を指すのか、の定義付けを後押しするものである。透明性により、そのような慣例が顧客のニーズや、脅威アクターの戦術手法と経済原理の変化、技術の進歩に合わせて時とともに変化することが促される。また、透明性はリソースがより限られているメーカーが、より成熟していて能力の優れたリソースを持つメーカーから学ぶことを後押しする。情報共有についての会話は、リアルタイムの脅威指標だけでなく、以下に挙げる要素も含めて行われるべきである：

透明性により必然的に、セキュリティ関連の意思決定が開発プロセスの早い段階で行われるように、そしてそうした意思決定をエンジニアやセキュリティ専門家だけでなく経営陣も継続的に行うようになる。透明性は、製品に説明責任を組み込むものである。

ここで、「透明性」の前に「徹底的な」という形容詞を選んで使用していることについて特記しておく。今日、メーカーがソフトウェアをどのように開発・保守しているか、そして経時的なデータを使ってプログラムをどのように成熟させているかについての詳細な情報を公開することは一般的ではない。ソフトウェア業界では、どのようにソフトウェアを設計しているかのガイドツアーを提供するメーカーはほとんどない。ソフトウェアメーカーが他社のSDLCプログラムがどのように構成され、そうしたプログラムが顧客環境下で現実の攻撃者に対してどの程度の耐性を発揮しているのかを目にする機会に恵まれることは、稀である。セキュリティ上の欠陥に対するコスト測定のため、そして複数クラスの脆弱性を排除するための戦略などのトピックについての情報共有をより活発にすることは、業界全体の共通した利益となるはずである。こうした一般的な慣行の結果、すべてのソフトウェアメーカーが、製品セキュリティへの対処法を自ら身に付けることとなるだろう。おそらく、セキュリティ機能に贅沢品として追加料金をかけなくなれば、安全とセキュリティはプロフィットセンターではなくコストセンターとなるため、企業は相互協力や透明性の確保・推進による負担軽減からのメリットが得られるのではないか。

我々は、ソフトウェア業界の進化を著しく加速させる戦術手法に焦点を当てたい。これまでのような場当たり的かつ漸進的な改善を今後も続けていく余裕はもうない。我々がインテリジェントかつ適応力のある敵対者からの脅威を皆で克服するためには、今日では多少不安なものであっても業界を前進させるようなレベルの透明性を確保しなければならない。メーカーの中には、今日の時点でこれらのセキュアバイデザイン原則のいくつかを体現している業者もある。ウィリアム・ギブスンが言ったように、「未来は既にここにある、ただあまり均等に配分されていないだけ」なのだ。**徹底的な透明性はその情報の配分を助け、敵対者よりも守る側の者に利益をもたらす。**

透明性がもたらすものは、各メーカー同士のSDLC成熟の後押しにとどまらない。潜在的な顧客や投資家は、メーカーが実行した投資やトレードオフについて、そしてそうした投資により顧客のために作り上げられたセキュリティ態勢についても、より多く学ぶことができる。徹底的な透明性を確保・推進するメーカーは、顧客が価格や機能だけでなくセキュリティについても考慮したうえで購入についての意思決定を行えるように、顧客に情報を提供する。

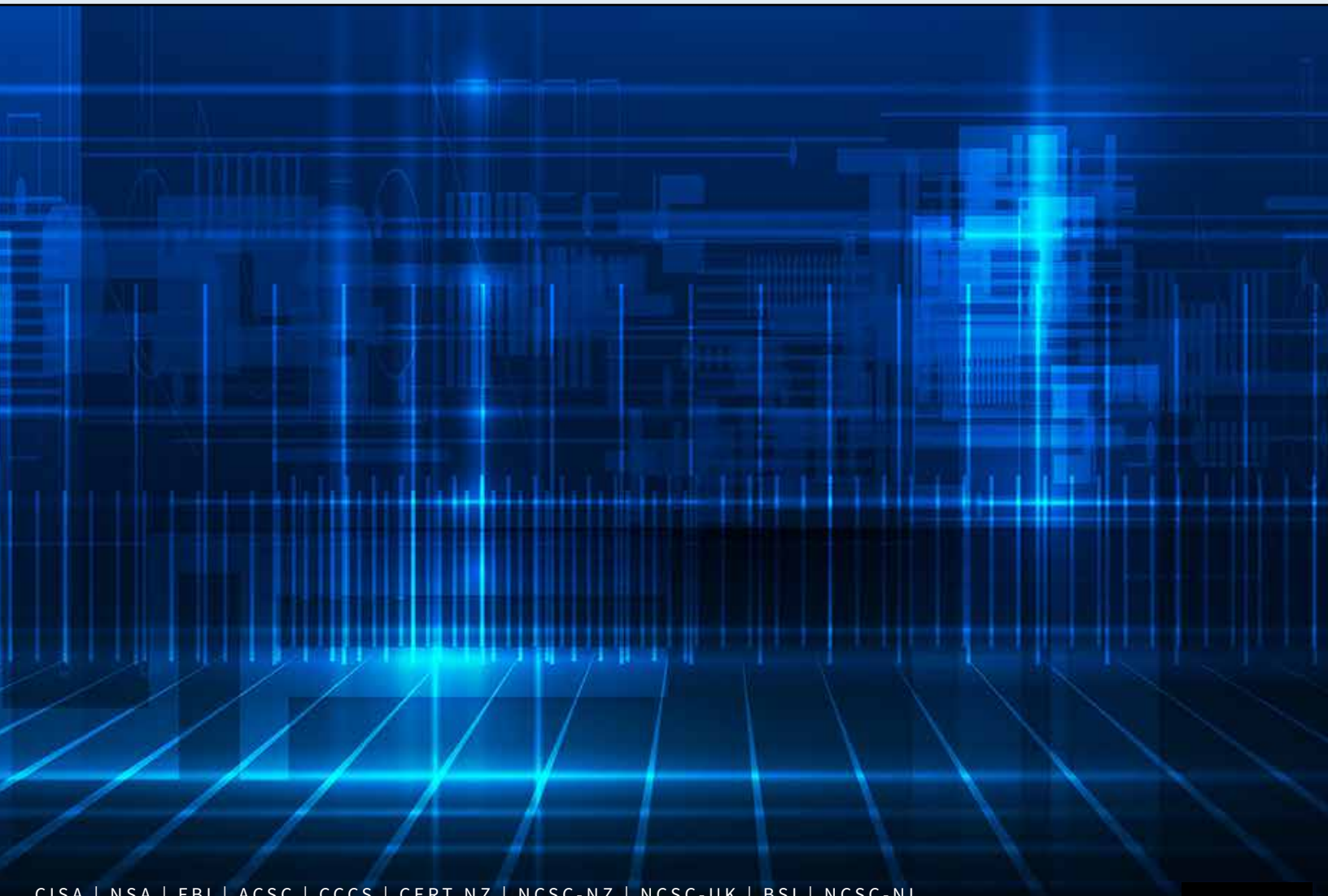
組織がサプライチェーンやSDLCを確保するために精一杯努力しているにもかかわらず、メーカーはここ最近、ビルドプロセスへの侵入等の被害を受けている。徹底的な透明性を確保・推進すれば、発生した攻撃そのものに加えて、企業が将来的な攻撃の予防と検知のために行った改善内容を公表することにつながるはずである。このようなかたちでの情報共有は、他の組織が同様の被害に直面することなく学ぶための機会となるだろう。

この原則の実施を実証するために

この原則を実施していること実証するために、ソフトウェアメーカーは以下に挙げるものを含めた様々な手順を取るべきである：

セキュアバイデフォルトの慣行

1. **セキュリティ関連の総計的な統計と傾向を公表する。**公表される情報のトピックの例としては、顧客や管理者によるMFA採用についてのデータや、安全ではないレガシープロトコルの使用についてのデータなどが挙げられる。
2. **パッチ適用の統計を公表する。**製品の最新バージョンを使用している顧客の割合と、アップデートをより容易で信頼性の高いものにするために自社が行っている取り組みを、詳しく説明すること。
3. **未使用の管理者特権のデータを公表する。**過剰な権限付与についての顧客全体の集計情報と共に、攻撃対象領域を減らすために自社製品に導入しているナッジングその他の変更も合わせて公表する。こうした未使用特権は、シートベルトチャイムのような、管理者に対するアラート発信の対象として適した項目の候補となる可能性が高い。



セキュアな製品開発慣行

1. **内部セキュリティ統制を確立する。**多くの企業が、自社データをクラウドサービス事業者に移すメリットを目の当たりにして来た。しかし、今やそのクラウド事業者が攻撃者の標的となっている。サービスとしてのソフトウェア（SaaS）事業者は、自らの内部統制についての統計を公表するべきである。たとえば、SaaS事業者はFast Identity Online（FIDO）認証のようなフィッシングに耐性のあるMFAの内部導入に関する統計を公表するべきであろう。理想的には、社内のどのスタッフも、フィッシング耐性のあるMFAで認証を受けなければ顧客その他の機密データにアクセスすることはできない、と言える状態にあることが望ましい。
2. **ハイレベルな大枠の脅威モデルを公表する。**セキュアバイデザインの製品は、それをつくる者が何を誰から守ろうとしているのかを記述した、脅威モデル文書の作成から始まる。効果的な脅威モデルとは、侵入が実環境下でどのように発生しているのかの十分な情報に基づいたものであり、エンタープライズ環境と開発環境の双方、さらにはソフトウェアメーカーが意図する顧客環境下での使用方法に対応しているべきである。
3. **安全なSDLCの詳しい自己認証を公表する。**NIST SSDFもしくはその他の同様の枠組みに従っているメーカーは、成熟したソフトウェア開発ライフサイクルに向けて積極的に活動している。メーカーが、どの制御措置をどの製品に対して実施したのかについての自己認証を公表することは、上記のベストプラクティスを順守し、顧客に対してより高い信頼を提供するというコミットメントを示すことになる。なお、他の認証スキームとしては、たとえば「Israel Cyber Supply Chain Methodology［イスラエル・サイバーサプライチェーン・メソドロジー］」などがある。
4. **脆弱性についての透明性を確保・推進する。**特定された製品の脆弱性を正確かつ完全なかたちのCVEエントリーとして公開するよう徹底するというコミットメントを公表する。こうした姿勢は、脆弱性の根本原因を特定・識別するCWEの入力項目についてはとりわけ徹底が求められる。公表・共有されているCVEデータベースの正確性と完全性がより高いほど、業界は製品がどの程度より安全になっていて、どのクラス属性の脆弱性が最も蔓延しているのかを、より適切に追跡管理することができるようになる。ただし、CVE件数は健全なコード分析とテストコミュニティが存在する証でもあるので、CVE件数をネガティブな指標としてみなしてしまう誘惑には注意が必要である。メーカーによるセキュアバイデザイン理念の導入が進む過程では、既存コードに存在する脆弱性の発見とその緩和措置がより広範に行われるため、初期段階の生データにおいては、CVE件数が増加する可能性がある。メーカーは、当該クラス属性の脆弱性全体に対処するために取られた措置やパターンを含む、過去の脆弱性に関する分析を公表するべきである。たとえば、ある企業のCVEの大部分がクロスサイト・スクリプティング（XSS）に関連するものである場合、根本原因の分析や、対応（例：XSSを防止するウェブテンプレートフレームワークへの移行）、それらの結果を文書化することにより、数十年前から緩和策が理解されているクラスの脆弱性の被害には遭わないであろうことを顧客に向けて発信することができる。
5. **ソフトウェア部品表（SBOM: Software Bills of Material）を公表する。**メーカーは、自らのサプライチェーンを掌握しているべきである。組織は各製品のSBOM [2]を作成・維持し、サプライヤーにデータを要求し、下流にいる顧客やユーザがSBOMにアクセスできるようにすることが望ましい。こうすることにより、自社製品の作成時に使用するコンポネントの理解に向けての誠実な努力や、新たに特定されたリスクに対応する能力を示すだけでなく、サプライチェーン内のモジュールのひとつに新たな脆弱性が発見された際にどのように対応すべきかを顧客が理解するのに

も役立つ。参考までに、日本の経済産業省（METI）は、「Guide of Introduction of Software Bill of Materials (SBOM) for Software Management」［ソフトウェア管理に向けたSoftware Bill of Materials（SBOM）の導入に関する手引］を公開している。透明性確保の対象は、組み込みデバイスのファームウェアや、AI/機械学習(ML)で使用されるデータおよびモデルにまで拡大されるべきである。SBOM は、購入の意志決定と運用・実務能力を支援するだけでなく、インフラにおいて悪意のあるサプライチェーン攻撃を検出して対応するという重要な役割を果たしている。

6. **脆弱性開示ポリシーを公表する。**つぎの各項の内容を含む脆弱性開示ポリシーを公表する：(1) そのメーカーが提供する全製品に対するテストと、そのテストの条件を許可する、(2) 当該ポリシーに合致した行動を取っている際の法的な免責を提供する、(3) 決められた期間後に脆弱性の公開を認める。メーカーは、発見された脆弱性の根本原因分析を行い、実行可能な範囲で最大限に、その脆弱性クラス全体を排除するための措置を講じるべきである。基準言語については、CISA の「Vulnerability Disclosure Policy Template」を参照されたい。

セキュリティに能動的なビジネス慣行

1. **セキュアバイデザインの支持責任者となる経営陣上層部担当者を指名し、公表する。**多くの組織において、セキュリティは（品質と同様に）技術チームに委ねられており、そうしたチームに与えられた製品のセキュリティを劇的に改善させるための構造改革を行う能力は、限定的なものである。セキュアバイデザインのプログラムを監督する経営幹部を指名しこれを公表することで、製品セキュリティはトップレベルの経営問題に変貌する。

2. **セキュアバイデザインに向けてのロードマップを公表する。**メーカーは、フィールドテスト・レポートの詳細や脆弱性クラス全体を排除するために取られた措置、そして他の原則に挙げられている項目を含めて、顧客のセキュリティを改善するためにSDLCに加えられた各変更点を文書化するべきである。セキュリティ改善プログラムには、品質改善への取り組みと同様に、計画、管理、改善というそれぞれ明確に異なる3つの段階がある。語るよりも示すという精神に基づき、ロードマップと各段階の詳細を公表することで、当該製品がセキュアバイデザインであるという自信や信頼が構築される。意義のある重要な進歩を達成したときには、それを透明性レポートに詳述できる。こうすることで自らのセキュアバイデザイン原則へのコミットメントを示すだけでなく、実在する証拠を示すことにより、他のメーカーを刺激し同様のプログラムを採用するよう奨励することもできる。

3. **メモリ安全性のロードマップを公表する。**メーカーは、メモリセーフな言語を用いて既存製品を移行させたり新製品を構築したりすることにより、脆弱性クラスの中でも最大規模のものを排除する措置をとることができる。このような取り組みはすべてのケースに適用できるものではないかもしれないが、メーカーはアプリケーション全体を作成し直す代わりに、メモリセーフな言語でのラッパーアプリケーションの開発を検討することもできるだろう。また、このロードマップには、メーカーが採用や訓練・研修、コードレビューその他の内部プロセスをどのように更新しているのかや、オープンソースコミュニティが同様の取り組みを進められるようメーカーがどのように支援しているのかも含むことができる。

4. **結果を公表する。**セキュアバイデザインの理念を実現するために自らのSDLCを更新する中で、組織は即効性のある成果や、成功のためにより多くのリソースを要するもの、そしていくつかの予期しないつまずきや後退があることを学ぶだろう。組織内の成功事項と障害物を提示すれば、こうした結果から業界全体が学ぶことができる。

原則3：トップレベルから率先して取り組む

詳細説明

我々はこの理念全体を指して「セキュアバイデザイン」と呼んでいるが、顧客の安全のためのインセンティブは、製品設計段階のはるか前から始まっている。こうしたインセンティブは事業目標や、暗示的および明示的な目的、望ましい成果から始まっているのだ。経営陣上層部がセキュリティを経営上の優先事項と定めて社内インセンティブを与えたり、セキュリティを設計要件とする文化を会社全体で醸成したりして初めて、最良の成果を上げられるのである。

製品の安全には技術的な専門知識が不可欠であるが、安全は技術スタッフだけに委ねられる問題ではない。それは、トップから取り組みを始めなければならない経営上の優先事項である。

メーカーが最初の2つの原則を積極的に実施し、意味のあるアーティファクトをつくり上げているのであれば、第3の原則など本当に必要なのかと疑問に思う声もあった。企業がどのようにそのビジョンや使命、価値、文化を確立するのかは当該企業の製品に影響を与えるものであり、これらの各項目の相当な部分は、経営上層部に帰属するものである。こうしたトップレベルによる取り組みは、安全と品質を劇的に向上させてきた他の産業にも見られる。著名な品質専門家であるJ.M.ジュランは、つぎのように書いている：

> ***品質をリードする立ち位置につくには、上層部が各々個人的に、品質のための事業経営を担うことが求められる。品質をリードする立場を勝ち取った企業では、上層部が自らそのような取り組みを指導していた。私の知る限り、これがあてはまらない例外はひとつもない。[3]***

我々は、セキュリティは製品品質のサブカテゴリのひとつであると考えている。セキュリティと品質が、技術スタッフのみに任せられた単なる技術機能ではなく経営上の必須事項となったとき、企業は顧客のセキュリティニーズにより迅速かつ効率的に対応できるようになるだろう。さらに、ソフトウエアセキュリティが初期段階から事業上の中核的な優先事項であるよう徹底すべく必要な投資をすることにより、ソフトウエアの欠陥に対処するための長期的なコストが削減され、ひいては国全体のセキュリティリスクも低減することとなるだろう。

経営陣上層部が企業の社会的責任（CSR）プログラムを導入したときと同じように、ソフトウェアメーカーを含む企業の取締役会はサイバーセキュリティ・プログラムを先導するためにより積極的・能動的な役割を果たすべきであるという認識が、高まっている。この新しい考え方を指して、「企業のサイバー責任（CCR: Corporate Cyber Responsibility）」という用語が使われることもある。

この原則の実施を実証するために

この原則を実施していることを実証するために、ソフトウェアメーカーは以下に挙げるものを含めた様々な手順を取るべきである：

1. **企業の財務報告に、セキュアバイデザイン・プログラムの詳細を含める。**メーカーが上場企業の場合は、年次報告内にセキュアバイデザインに関する取り組みだけを扱った項目を設ける。自動車会社の年次財務報告には、中央集権型および分散型の品質・安全委員会に関する情報を含む、ドライバーと同乗者の安全に関する項目を設けていることが一般的である。財務報告にセキュアバイデザイン・プログラムの詳細を記載することで、その組織が顧客のセキュリティと企業の財務成果を結びつけて考えており、単に流行しているからという理由でマーケティング資料の用語を取り上げているのではないと示すことができる。
2. **自社の取締役会に向けて、定期報告を行う。**最高情報セキュリティ責任者（CISO）からの企業の取締役会への報告には通常、現在および計画中のセキュリティ・プログラムや、脅威、実際にあったセキュリティインシデントおよびセキュリティインシデントが疑われる案件、その企業のセキュリティ上の態勢と健全性にまつわるその他の最新情報が含まれる。取締役会は、自社のセキュリティ態勢に関する情報を得ることに加え、製品セキュリティおよびそれが顧客のセキュリティに及ぼす影響についての情報も報告するよう要請するべきである。また、取締役会は顧客のリスク低減を進めるためにCISOのみに頼るのではなく、まず経営陣の他の面々にもその責任を果たすよう求めるべきである。
3. **セキュアバイデザイン担当経営幹部の権限を強化する。**技術チームが「経営陣の理解・支持（Executive Buy-In）」を得ている組織と、経営陣上層部が自ら標準的なビジネスプロセスを用いて顧客のセキュリティ改善プロセスを指揮している組織とでは、著しく大きな違いがある。「経営陣の理解・支持」という用語は、［英語ではBuy-Inと言われることから］、顧客のセキュリティプログラムというアイデアを誰かが売り込まなくてはならなかったこと、つまり顧客のセキュリティはトップレベルの経営目標ではなかったことを示唆している。担当経営幹部は、顧客セキュリティ上の成果を達成するための製品投資に影響を与えるための権限を付与されなくてはならない。
4. **意味のある社内インセンティブを設ける。**不健全なインセンティブを設けないよう留意しつつ、顧客のセキュリティ向上のために、他の価値ある行動や業績と釣り合うものとなるよう報酬制度を整備する。セキュアバイデザイン担当の経営幹部から、製品管理、ソフトウェア開発、サポート、営業、法務その他の各部門に至るまで、顧客のセキュリティ向上に関するインセンティブを、採用や昇進、給与、賞与、ストックオプション、そして業務上のその他のプロセスに組み込む。たとえばソフトウェア開発者の昇進要件を設定する際に、アップタイムやパフォーマンス、機能向上などの他の要件とともに、製品セキュリティの改善も考慮の対象事項に入れるようにする。
5. **セキュアバイデザイン協議会を設置する。**一部の業界では、品質協議会を中央に設置し、主要部門や事業ユニットに品質担当者を配置することが一般的に行われている。こうしたグループは、中央集権的・分権的双方の人員を含めることで、組織の深部からテレメトリ情報を得ながら、トップレベルの目標に向けた品質改善に取り組んでいる。これと同様に、セキュアバイデザイン協議会が設けられていれば、組織全体を通してセキュアバイデザイン目標に向けてセキュリティ改善を進めるものとなるだろう。
6. **顧客協議会を設立し、発展させる。**多くのソフトウェアメーカーでは、地域や業種、規模の異なる顧客たちで構成された顧客協議会を設けている。このような協議会は、自社製品の導入に伴う顧客の成功例や課題について非常に多くの情報をもたらすことができる。協議会の議題は、仮にその時点での参加者の最大関心事項ではなかったとしても、顧客の安全を取り上げそれに特化したトピックで構成するようにする。なお、顧客協議会からの報告先はどこなのか、また、参加者からどのようにして導入された製品のセキュリティに関する知見を引き出すのかを検討すること。たとえば、協議会がマーケティングや営業目的、または製品管理目的に偏っていないかどうかを考慮するべきである。また、セキュアバイデザイン担当の経営幹部は、このような顧客との交流の内容促進・活発化を支援し、フィールドスタディーのような、本ガイド内に記されている他の要素とも関連付けるべきである。

セキュアバイデザインの戦術手法

米国標準技術研究所（NIST: National Institute of Standards and Technology）SP 800-218としても知られているSecure Software Development Framework（SSDF）は、ハイレベルな大枠を示した一連の中核となるセキュアなソフトウェア開発慣行であり、こうした慣行はソフトウェア開発ライフサイクル（SDLC）の各段階に統合できるようになっている。これらの慣行を実践することで、ソフトウェアメーカーはリリースされたソフトウェアの脆弱性をより効果的に発見・除去したり、脆弱性の悪用に関する潜在的な影響を軽減したり、脆弱性の根本原因に対処して将来的な再発を防止することに役立てられる。

本書作成機関は、SSDF慣行を参照している原則を含め、セキュアバイデザインの戦術手法を使用するよう推奨している。ソフトウェアメーカーは、自社製品ポートフォリオ全体でセキュアバイデザインのソフトウェア開発慣行をより多く取り入れるために、書面によるロードマップを作成するべきである。以下は、網羅的ではないがロードマップのベストプラクティスのリストである：

- **メモリセーフなプログラミング言語の使用（SSDF PW.6.1）。**可能であれば必ず、メモリセーフな言語を優先的に使用する。本書作成機関は、メモリに特化した緩和策がレガシーコードベースに対する短期的な戦術として有効である可能性があることを認める。これにはたとえば、C/C++言語の改善や、ハードウェア緩和策、アドレス空間配置のランダム化(ASLR)、制御フローの整合性(CFI)、ファジングなどが含まれる。それでもなお、メモリセーフなプログラミング言語の採用はこうしたクラスの不具合を撤廃することができるというコンセンサスが高まっており、ソフトウェアメーカーはこうした言語を採用する方法を探るべきである。最近のメモリセーフな言語としては、C#、Rust、Ruby、Java、Go、Swiftなどが挙げられる。詳細については、NSAのメモリ安全性に関するインフォメーションシートを参照されたい。

- **セキュアなハードウェア基盤。**Capability Hardware Enhanced RISC Instructions（CHERI）で説明されているような、従来のハードウェア命令セットアーキテクチャ（ISAs）を拡張できるきめ細かなメモリ保護を可能にするアーキテクチャ機能と合わせて、信頼されたプラットフォームモジュール（TPM）やハードウェアセキュリティ・モジュールなど、その他の機能を取り入れる。詳細は、ケンブリッジ大学のCHERIウェブページを参照されたい。

- **セキュアなソフトウェアコンポーネント（SSDF PW 4.1）。**消費者向けソフトウェア製品における強固なセキュリティを徹底するために、検証された商用、オープンソースおよびその他のサードパーティー開発者による、十分に安全なソフトウェアコンポネント（例：ソフトウェアライブラリ、モジュール、ミドルウェア、フレームワークなど）を調達し、維持すること。

- **Webテンプレートフレームワーク（SSDF PW.5.1）。**クロスサイトスクリプティングなどのウェブ攻撃を避けるため、ユーザ入力の自動エスケープを実装するウェブテンプレートフレームワークを使用すること。

- **パラメータ化されたクエリ(SSDF PW 5.1)。** クエリにユーザ入力を含めることがない、パラメータ化されたクエリを利用することでSQLインジェクション攻撃を避ける。

- **静的、動的アプリケーションセキュリティのテスト（SAST/DAST）** (SSDF PW.7.2, PW.8.2)。これらのツールを使用して製品ソースコードおよびアプリケーションの動作を分析し、エラーが発生しやすい慣行を検出する。これらのツールは、不適切なメモリ管理からエラーが発生しやすいデータベースクエリ構成（例：ユーザ入力がエスケープされておらず、SQLインジェクション攻撃を招いてしまう)といったものまで、様々な問題に対応している。SASTおよびDASTツールは開発プロセスに組み込むことができ、ソフトウェア開発の一環として自動的に実行することができる。なお、SASTおよびDASTツールは、製品が期待されているセキュリティ要件に準拠しているよう徹底するために、単体テストや結合テストなどの他の種類のテストを補完するかたちで利用されるべきである。問題が特定された場合、メーカーは脆弱性に体系的に対処するために根本原因を分析するべきである。

- **コードレビュー** (SSDF PW.7.1, PW.7.2)。製品に組み込まれたコードが、他の開発者によるピアレビューや「エラー埋め込み（バグ埋め込み）」などの品質管理手法の適用を経るよう徹底することを目指す。
- **ソフトウェア部品表（SBOM: Software Bill of Materials)** (SSDF PS.3.2, PW.4.1)。製品に組み込まれるソフトウェアセットの可視化をもたらすために、SBOM[4] の作成を組み込む。
- **脆弱性開示プログラム**(SSDF RV.1.3)。セキュリティ研究者が脆弱性を報告し、その報告行為に対する法的責任から免除されることを可能にする、脆弱性開示プログラムを確立する。その一環として、サプライヤーは発見された脆弱性の根本原因を見極めるためのプロセスも確立するべきである。そのようなプロセスには、この文書に記されているセキュアバイデザインの慣行(または類似の慣行)が採用されていれば、脆弱性の混入を妨げていたのかどうかの見極めが含まれているべきである。
- **CVEの完全性。**公開されたCVEには、ソフトウェアセキュリティ設計上の欠陥を業界全体で分析できるように、根本原因やCWEが含まれるよう徹底する。すべてのCVEが正しく完全であるよう徹底確認すると追加時間を要する可能性があるが、こうすることで多様な団体・組織や個人が、すべてのメーカーと顧客に利益をもたらすような業界トレンドを把握できるようになる。脆弱性管理の詳細については、CISAのStakeholder-Specific Vulnerability Categorization（SSVC）[ステークホルダーに特化した脆弱性のカテゴリ分類] ガイダンスを参照されたい。
- **縦深防御。**インフラ設計は、単一のセキュリティ制御の侵害がシステム全体の侵害につながるようなことがないようなかたちにする。たとえば、利用者権限の厳密な付与や、アクセス制御リストの導入を徹底することにより、侵害されたアカウントによる影響を軽減できる。また、ソフトウェアのサンドボックス環境化技術は脆弱性の隔離を可能にし、アプリケーション全体の侵害を限定的なものに抑えることができる。
- **サイバーパフォーマンス目標を満たす。**基本的なセキュリティ慣行条件を満たす製品を設計する。CISAのサイバーパフォーマンス目標（CPG：Cyber Performance Goals）は、組織が実施すべき基本的なベースラインとなるサイバーセキュリティ策の概要を示している。加えて、組織の態勢を強化するためのその他の手段については、CISAのCPGとの類似点も見られる英国のCyber Assessment Frameworkを参照されたい。メーカーがCPGを満たしていない場合は— すべての従業員に対してフィッシング耐性のある多要素認証を要求していないなど— そのメーカーはセキュアバイデザインな製品を提供しているとはみなされない。

本書作成機関は、これらの変更が組織態勢の大幅な変更であることを認識している。そのため、この要件を満たすための変化の導入は、各組織にカスタマイズされた脅威モデリングや、重大性、複雑さ、そしてビジネスへの影響に基づいて優先順位を付けることが望ましい。これらの慣行は新しいソフトウェアに導入し、追加のユースケースや製品にも対応するよう段階的に拡張していくことができる。場合によっては、特定製品の重大性とリスク態勢から、こうした慣行の導入スケジュールを加速させる価値があると判断されるかもしれない。あるいは、一連の慣行をレガシーコードベースに導入したうえで、時間をかけて修正すれば良いという場合もある。

[4] 本書作成機関の一部は、セキュリティ保証を得るための別のアプローチを、ソフトウェアサプライチェーン周りで模索している。

セキュアバイデフォルトの戦術手法

本書作成機関はソフトウェアメーカーに対して、セキュアバイデザインに沿った開発慣行の採用に加えて、自社製品におけるセキュアバイデフォルトの設定を優先するよう推奨している。こうしたソフトウェアメーカーは、これらの慣行が改訂されるのに合わせてこうした各慣行に準拠するよう製品を更新する努力をすることが望ましい。以下に例を挙げる：

- **デフォルトパスワードを排除する。**製品は、広く一般に共有されているパスワードがデフォルトとして設定された状態で提供されるべきではない。デフォルトのパスワードを排除するために、本書作成機関ではインストールや設定の最中に製品側から管理者に強力なパスワードを使用すること、あるいは当該製品について各デバイスに固有で強力なパスワードが設定された状態で出荷されるよう推奨している。

- **上級権限ユーザに対する多要素認証（MFA）の必須化。**我々は、エンタープライズにおけるソフトウェア導入が、自らのアカウントをMFAで保護していない管理者により管理されていることが多いのが現状であることを目にしている。管理者が価値の高い攻撃ターゲットであることを考えると、製品はMFAをオプトイン項目ではなく、オプトアウト項目にするべきである。さらに、システムは管理者が自らのアカウント上でMFAを有効にするまで、MFAに登録するよう定期的に促すようにするべきである。オランダのNCSCは、CISAのものと類似したガイダンスを有している。詳細については、MFAの Mature Authentication Factsheet［成熟した認証に関するファクトシート］を参照されたい。

- **シングルサインオン（SSO）。**ITアプリケーションは、最新のオープンスタンダードを介してシングルサインオンを実装するべきである。例としては、セキュリティアサーションマークアップ言語（SAML: Security Assertion Markup Language）やOpenID Connect(OIDC)などが挙げられる。この機能は、追加料金なくデフォルトで使用可能とすることが望ましい。

- **安全なログ記録。**高品質の監査ログを、追加料金・追加設定なしで顧客に提供する。監査ログは、潜在的なセキュリティインシデントを検知して対応をエスカレーションするために不可欠なものである。また、同ログは疑いのある、もしくは既に確認されたセキュリティインシデントの調査の際にも不可欠なものである。協定世界時(UTC)や標準タイムゾーン・フォーマット、そして堅牢な文書化技術を使用するアプリケーションプログラミングインタフェース(API)アクセスを有するセキュリティ情報イベント管理システムとの容易な結合（統合）を提供するなどの、ベストプラクティスを検討する。

- **ソフトウェア認証プロファイル。**ソフトウェアサプライヤーは、認証されたプロファイルが与えられる役割およびその指定されたユースケースに関する推奨事項を示すべきである。メーカーは、推奨されたプロファイル認証から顧客が逸脱した場合にリスクが増大することを顧客に通知する、目に見える警告を含めることが望ましい。例としては、医師はすべての全患者記録を閲覧できるが、医療スケジュール管理者は、予約のスケジューリングに必要な一定の情報のみにアクセスできるよう制限される、などが考えられる。

- **後方互換性ではなく、将来に目を向けたセキュリティ。**製品セキュリティにリスクをもたらすにもかかわらず、後方互換性のあるレガシー機能が製品に含まれることがあまりに多く、しかもそうした機能は有効になっていることが少なくない。後方互換性よりもセキュリティを優先し、それが破壊的な変更を引き起こすことになっても、安全ではない機能を削除する権限をセキュリティチームに与えるべきである。

- **「堅牢化ガイド」の改訂を追跡管理し、サイズを縮小する。**製品に含まれる「堅牢化ガイド」のサイズを縮小し、時間の経過に従いソフトウェアの新しいバージョンがリリースされる度に、ガイドのサイズも縮小してくよう徹底する。「堅牢化ガイド」の記載内容を、製品のデフォルト設定として統合する。本書作成機関は、

短くなった堅牢化ガイドは既存顧客との継続的なパートナーシップによりもたらされるものであり、そこにはユーザ体験(UX)を含む多くの製品チームによる努力があると認識している。

- **セキュリティ設定のユーザ体験に対する影響を考慮する。**新しい設定は、そのひとつひとつがエンドユーザの認知面での負担を増大させるため、設定変更によって得られるビジネス上のメリットと合わせて評価されるべきである。設定など存在しないことが理想であり、最も安全な設定はデフォルトで製品に統合されているべきなのである。設定が必要な場合は、デフォルトの選択が共通の脅威に対して広く安全でなければならない。

本書作成機関は、これらの変更がソフトウェアの導入・活用方法に業務・運用上の影響を与える可能性があることを認識している。したがって、運用上とセキュリティ上の考慮のバランスをとるために、顧客の意見が不可欠である。我々は、これらのアイデアを組織の最重要製品に優先して導入するという書面によるロードマップの作成と経営層からの支援を育むことが、セキュアなソフトウェア開発慣行に移行するための第一歩であると確信している。顧客からの意見は重要であるが、我々は、改善された基準、特にネットワークプロトコルを取り入れる意志が顧客にない場合や、そうした導入が不可能であるという重大な事例を目にしてきた。メーカーが顧客に対し、常に最新の状態を維持し、いつまでも脆弱な状態なままでいないよう促すための、意味のあるインセンティブを設けることが重要である。

堅牢化ガイドか、緩和ガイドか

堅牢化ガイドは、セキュリティ制御が製品開発開始当初から製品アーキテクチャに組み込まれていないことで発生し得るものである。このため堅牢化ガイドは、攻撃者が安全でない機能を特定し悪用するためのロードマップとなってしまう可能性も抱えている。多くの組織において、堅牢化ガイドの存在が知られていないこともよく見られ、機器の設定が安全でない態勢のまま放置されてしまうことになる。こうした堅牢化ガイドの代わりに、逆転の発想によるモデルである「セキュリティ緩和ガイド」と呼ばれるものが導入されるべきであり、この緩和ガイドにより、どのセキュリティ構成を変更すべきなのかが、その変更によりもたらされるセキュリティリスクのリストと共に説明されるのが望ましい。このようなガイドは、正しく使用される可能性が高まるように、分かりやすい言葉で、トレードオフも説明できるセキュリティ実務者によって書かれることが望ましい。

本書作成機関はソフトウェアメーカーに対し、製品を保護する方法を列挙した堅牢化ガイドを作成するのではなく、セキュアバイデフォルトのアプローチに転換し、「緩和ガイド」を提供することを推奨する。緩和ガイドでは、各意思決定に対するビジネスリスクが平易な分かりやすい言葉で説明され、悪意あるサイバー侵入攻撃のリスクに対する組織の意識向上を促すこととなる。セキュリティのトレードオフは、セキュリティとそれ以外のビジネス上の要件とのバランスを図りながら、顧客企業の経営陣上層部が決定すべきことである。

顧客への推奨事項

本書作成機関は、顧客組織が利用しているソフトウェアの供給メーカーに対して、その製品がもたらす安全上の結果の説明責任を持たせるよう推奨する。その一環として、本書作成機関は顧客組織の経営陣に対し、セキュアバイデザインかつセキュアバイデフォルトである製品の購入を優先するよう推奨する。こうした慣行は、ソフトウェアの購入前にそのセキュリティを評価することをIT部門に義務付けること、そして必要に応じて拒否する権限をIT部門に与えることを盛り込んだ、社内方針の策定で具体化することができる。IT部門には、セキュアバイデザインかつセキュアバイデフォルトの慣行（この文書で概説されているものと顧客組織が築いたもの双方を含む）を重視する調達要件を策定する権限が与えられるのが望ましい。さらに、ソフトウェア購入の際にIT部門が上記のような要件の適用を主張する場合、経営陣はそれを支持するべきである。組織として、特定のテクノロジー製品に関するリスクを受け入れるという意志決定をする場合には、正式に文書化し、経営幹部が承認し、定期的に取締役会に報告されるべきである。

企業ネットワークやID認証アクセス管理、セキュリティ業務、対応能力などの、エンタープライズ組織のセキュリティ態勢を支える重要なITサービスは、ビジネス上の重要機能とみなし、当該組織の目的達成に対する重要性に応じて予算拠出が行われるのが望ましい。顧客組織はこうした能力を強化するための計画を策定して、セキュアバイデザインやセキュアバイデフォルトの慣行を積極的に推進するメーカーを利用すべきである。

顧客組織は、可能な限り自社の主要ITサプライヤーと戦略的関係を構築するよう努力するべきである。こうした戦略的関係は組織内の複数層におよぶ信頼関係を含み、問題解決や共通の優先事項の特定などを進める役割を果たすものである。セキュリティはこのような関係の重要な要素となるべきであり、各顧客組織は関係の正式チャンネル（契約やベンダー契約など）とインフォーマルなチャンネルの双方において、セキュアバイデザインとセキュアバイデフォルトの慣行の重要性の強調に最善を尽くすべきである。顧客組織はテクノロジーサプライヤーに対して、内部統制に関する態勢や、セキュアバイデザインとセキュアバイデフォルト慣行の導入に向けてのロードマップに関する透明性を期待するべきである。

IT部門幹部はセキュアバイデフォルトを組織内の優先事項とするだけでなく、どの製品やサービスがこうした設計原則を体現しているのか理解できるよう、業界内の同業者と協力するのが望ましい。業界内のこうした幹部たちは、メーカーが今後の顧客のセキュリティ面での取り組みに優先して対応するよう、要望を調整することが望ましい。このように協力することにより、顧客はメーカーに対して意味のあるインプットを提供することができるようになり、メーカーがセキュリティを優先することにインセンティブをもたらすことができる。

また、クラウドシステムを利用する場合には、顧客組織はこうしたテクノロジーサプライヤーとの責任共有モデルというものを確実に理解しなければならない。つまり顧客組織は、顧客側の責任だけではなく、サプライヤー側のセキュリティに対する責任も明確に理解しておくべきである、という意味である。

顧客組織はクラウド事業者の中から、自らのセキュリティ態勢や内部統制、そして責任共有モデルに基づく義務を果たす能力について、透明性の高い企業を優先することが望ましい。

免責事項

参考文献

CISA（米国サイバーセキュリティ・インフラストラクチャーセキュリティ庁）

- CISA's SBOM Guidance
- CISA's Cross-Sector Cybersecurity Performance Goals
- Guidelines on Technology Interoperability
- CISA and NIST's Defending Against Software Supply Chain Attacks
- The Cost of Unsafe Technology and What We Can Do About It | CISA
- Stop Passing the Buck on Cybersecurity: Why Companies Must Build Safety Into Tech Products (foreignaffairs.com)
- CISA's Stakeholder-Specific Vulnerability Categorization (SSVC) Guidance
- CISA's Phishing Resistant MFA Fact Sheets
- Cyber Guidance for Small Businesses | CISA

NSA（米国国家安全保障局）

- NSA's Cybersecurity Information Sheet on Memory Safety
- NSA's ESF Securing the Software Supply Chain: Best Practices for Suppliers

FBI（米国連邦捜査局）

- Understanding and Responding to the SolarWinds Supply Chain Attack: The Federal Perspective
- The Cyber Threat - Response and Reporting
- FBI's Cyber Strategy

National Institute of Standards and Technology (NIST)

- NIST's Digital Identity Guidelines
- NIST's Cyber Security Framework
- NIST's Secure Software Development Framework (SSDF)

豪サイバーセキュリティセンター (ACSC)

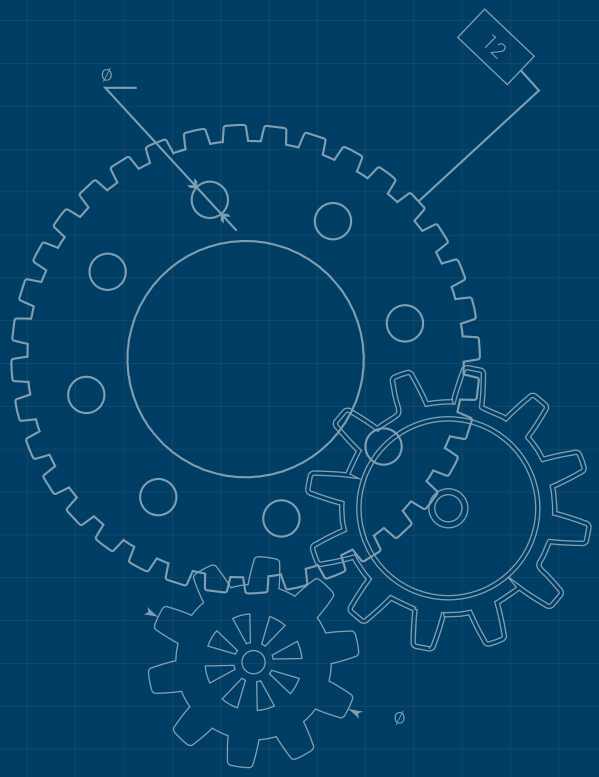
- ACSC's IoT Code of Practice Guidance for Manufacturers

英国国家サイバーセキュリティセンター

- The UK's Cyber Assessment Framework
- The UK NCSC's Secure Development and Deployment guidance
- The UK NCSC's Vulnerability Management guidance
- The UK NCSC's Vulnerability Disclosure Toolkit
- University of Cambridge's CHERI
- So long and thanks for all the bits - NCSC.GOV.UK

カナダサイバーセキュリティセンター (CCCS)

- CCCS's Guidance on Protecting Against Software Supply Chain Attacks
- Cyber supply chain:An approach to assessing risks
- Canadian Centre for Cyber Security's CONTI ransomware guidance



ドイツ連邦セキュリティ庁 (BSI)

- The BSI Grundschutz compendium (module CON.8)
- The international standard IEC 62443, part 4-1
- State of IT-security in Germany report, 2022
- BSI practices of web application security

オランダ国家サイバーセキュリティセンター

- NCSC-NL's Mature Authentication Factsheet

日本国内閣サイバーセキュリティセンター (NISC)

- Japan's National Cybersecurity Strategy

日本国経済産業省 (METI)

- Guide of Introduction of Software Bill of Materials (SBOM) for Software Management
- Collection of Use Case Examples Regarding Management Methods for Utilizing OSS and Ensuring Its Security

シンガポール・サイバーセキュリティ庁

- Technical Advisory on Secure API Development
- CSA SingCERT Vulnerability Disclosure Policy
- CSA SingCERT Incident Response Checklist
- CSA SingCERT Incident Response Playbooks
- CSA Security by Design Framework
- CSA Security by Design Framework Checklist
- CSA Guide to Cyber Threat Modelling
- CSA Cybersecurity Labelling Scheme

その他

- How Complex Systems Fail
- The New Look in complex system failure

参照文献

[1] https://csrc.nist.rip/publications/history/ande72.pdf

[2] https://www.cisa.gov/sbom and SBOMs references in TR 03183-2 https://www.bsi.bund.de/dok/TR-03183

[3] Juran on Quality by Design by J.M. Juran, 1992.