# ソフトウェアの安全性を意識した管理体制(ver.1.0)

～ソフトウェアは「ゆりかご」から「墓場」まで～

SOFTWARE ISAC

# 内容

## １．はじめに

情報技術の発展や活用によって時代は大きく変化しています。最近では「Society5.0[1]」といった新たな社会の実現に向けて「Internet of Things（モノのインターネット化）」や「デジタルトランスフォーメーション（DX）」など、デジタル技術の活用が広く進み、私たちの生活をより豊かにしていこうとしています。この変化は人間が生み出した技術があるからこそ実現できるものであって、ハードウェア（目に見えるシステムの物理構造）もソフトウェア（目に見えない指令や制御機能）も進化することによって実現できています。つまりいずれの発展も欠かすことはできません。

さて、そのソフトウェアはハードウェアと異なります。ハードウェアは人間でいう骨や筋肉であり、（特別な事情を除いて）簡単には衰えないよう強固に作られています。しかし、ソフトウェアは人間でいう脳の指令や病気やけがを治す免疫機能のように、より柔軟かつ迅速に判断できるように、常に変化できる仕組みになっています。つまり、ソフトウェアは更新や変化することが前提で作られており、この柔軟性と対応力こそ DX の根幹とも言えます。そのため、ソフトウェアの世界は目まぐるしく変化し、新たな開発や新たなインシデントも発生しています。

またソフトウェア開発の現場ではオープンソースソフトウェア（OSS）のようないわばハードウェアでいうところの「部品」を外部から調達して開発が行われることが標準的になっています。この OSS に後日脆弱性が発見され外部からの攻撃リスクが生じたり、開発した組織においても維持、管理することに課題があったりするなど、ソフトウェアの開発環境や部品、管理、維持する体制など、様々な課題が生じています。

さらにソフトウェア開発の管理体制を複雑にしているのが、日本の産業構造ともいえる「多重下請け構造」です。昨今、「情報子会社問題」とも言われる子会社に情報システム関連の対応を委託しているように、ソフトウェアの開発は自組織だけで完結することは少ないのが現状です。委託、再委託、再々委託と開発がどんどん下請けに出され、開発にかかわるステークホルダー（利害関係者）が増えていきます。またこの委託についてもソフトウェア開発そのものを要件定義や契約などを適切に行わずに、丸投げしている組織も見られます。

そこで本書はソフトウェア開発を行う組織の管理者や責任者に向けて、ソフトウェアにおける課題と、ソフトウェア管理体制の構築方法を簡易的に示すとともに、組織規模にかかわらず、最低限取り組むべきソフトウェア管理の在り方について説明します。

---

[1] サイバー空間（仮想空間）とフィジカル空間（現実空間）を高度に融合させたシステムにより、経済発展と社会的課題の解決を両立する、人間中心の社会（Society）
https://www8.cao.go.jp/cstp/society5_0/

## 2．ソフトウェア管理体制の必要性

ソフトウェアを開発することは指令、制御するための命を吹き込むことと同様であり、「ゆりかごから墓場まで」面倒を見ていく必要があります。しかし、事業を継続するためにもソフトウェア開発は様々なプロジェクトを抱え、開発が終われば別の開発を行っていることが多く、生み出したソフトウェアを親のように最後まで面倒を見ることは（特に受託開発の場合）難しく、いわば「誕生」させればおしまいになってしまっている開発やプロジェクトも少なくありません。

そのような状況の中、ソフトウェアをとりまく環境は変化し、脅威が日々進化し続けています。例えば、ソフトウェアに起因するインシデントとして以下のように様々な攻撃が発生しています。

- 広く使われている製品の脆弱性を悪用した攻撃
- 開発されたソフトウェアで用いられた OSS の脆弱性を悪用した攻撃
- ソフトウェアアップデートのために配信したファイルに不正なコードが埋め込まれていた。
- アップデートのために作成したファイルが差し替えられていた。
- 開発時にマルウェアが混入してしまった。
- そもそも開発環境自体がマルウェアに汚染されていた。

**図 1　ソフトウェア関連の脅威・サイバー攻撃例**

サイバー攻撃やソフトウェアの脆弱性対応など、ソフトウェアを継続的に維持していくために、またソフトウェアの「ゆりかごから墓場まで」を実践するために、各組織でソフトウェアの安全な開発ライフサイクル「Secure Development Lifecycle（SDL）」を理解する必要があります。これは文字通り、ソフトウェアを生み出す「企画・設計」の段階から「廃棄」にいたるフェーズのすべてにおいて、安全面を配慮して考える、ソフトウェア開発を行う組織であれば切っても切り離すことのできない考え方です。


**図 2　安全なソフトウェア開発の流れ**

ソフトウェア開発を行う組織は生み出した責任を最後まで面倒見なければならず、この SDL を考え、ソフトウェアの維持や管理することが必要です。しかし、急に「管理を強化せよ」といっても受託する金額が増えるわけでもなく、開発エンジニアの給料を減らして管理に充てることもできないため、管理に追加の費用を発生させることが難しい現状もあります。新しい命（ソフトウェア）を生み出す組織として、また多重下請け構造の日本においては、より上位（委託元）の組織も一体となって、ソフトウェアを管理する体制を構築し、運用していく必要があります。

そこで、このようなソフトウェア開発の現場を踏まえながら、ソフトウェア管理体制に最低限求められることを整理します。

## ３．ソフトウェア管理体制の構築に向けて

ソフトウェアの管理体制で代表的なものが PSIRT（Product Security Incident Response(or Readiness)Team）です。ここでは FIRST が公開し、Software ISAC で翻訳を行った「PSIRT Services Framework」を参照しながら、ソフトウェア管理に必要な体制を記述します。

まずソフトウェアの管理を行う上で大切なことは、ステークホルダーの把握をはじめとした現状の把握です。確認するポイントは以下の通りです。

① 契約状況の把握

どの組織と契約を行い、契約の種類（委任、請負など）は何なのか、また契約の期間や損害賠償の上限に関する確認、自動更新の有無、さらには再委託の可否などを確認しましょう。万が一、契約がない場合には早々に対応し、契約がある場合でも不利な契約になっていた場合は契約改善のための交渉を行いましょう。開発したソフトウェアに脆弱性が発見された場合の連絡や対応の迅速性、費用負担、利用 OSS の脆弱性管理主体などもあらかじめ合意しておきましょう。どのような合意が必要かは、CSAJ が公開している「ソフトウェア出荷判定基準」を確認しましょう。

② プロジェクトの把握

ソフトウェアの開発案件が決まると、開発するためのプロジェクトが正式に発足することになります。しかし、多くのプロジェクトを抱える組織やエンジニアもいるため、プロジェクトごとに、情報を整理しておく必要があります。プロジェクトの根拠となっている契約は何なのか、開発の内容、責任者やメンバーを明確にした体制、各担当者の作業内容、各担当者の開発する環境、そして仕事と個人的な連絡先などを確認し、把握しておきましょう。

③ 開発環境や開発に使うソフトウェア情報の把握

Software ISAC の調査では開発で用いるソフトウェアや開発環境はプロジェクトまたはエンジニアに依存している割合が高く、組織として対応できていない現状があります。開発したソフトウェアを管理していくためには、開発に用いられた環境（IDE,統合開発環境）、使用した OSS などを把握しておく必要があります。また、ソースコードの共有や連携などに用いたツール（ex.GitHub）、さらにはどこからその情報（リポジトリ）を持ってきた（フォークした）のかなども把握しておく必要があります。具体的には次の通りです。

- 作成者
  - 作成組織・作成者、作成時期、作成者の各種情報（注意点やコメントなど）
- SBOM コンポーネントリスト
  - コンポーネント名、バージョン、コンポーネント提供者、識別子、ダウンロード場所、ライセンス関連（権利関係、サポート期間など）

図 3　ソフトウェア管理に必要な項目例

言わばどのような部品を使い、その部品はどこから入手し、その部品をどこで組み上げたのかという情報を把握しておく必要があります。なお、最近ではクラウドを利用してサービス展開をする場合も増え、クラウドサービス（本番環境だけでなく、ステージング、検証環境なども含む）のアカウント自体のアクセス権限の管理なども重要になっています。

さらに開発環境におけるセキュリティ対策も忘れてはなりません。まずは本当に必要なステークホルダーだけが利用できるようになっているか。認証の仕組みをきちんと取り入れているか（二要素認証や複雑なパスワードの設定などを含む）。端末のセキュリティ対策を適切に行っているかを確認し、組織を跨ぐ場合は、利用に当たってのポリシーやルール、そして設定を行ったうえで開発しましょう。（なお、開発環境の設定などについては「ランサムウェアからソフトウェア開発企業を守るためのガイドライン指針となる具体的な対策手法を提案」などをご参照ください。）

④ 関係者のスキルやリテラシーの把握

開発を行うのが「人」であれば、インシデントを起こすのも「人」です。特に開発に直結するステークホルダーについてはスキルやリテラシーを把握しておくことが大切です。スキルについてはこれまでの業務経験や資格などからある程度は把握することができるでしょう。リテラシーの把握は入社時に用いられる一般的な適性検査を実施し、その傾向を把握しておくことが望ましいでしょう。コストや時間をかけられない場合は、まずは会話をすることです。例えば、その会話の中で、これまで発生している「退職時の情報持ち出し」や「ソースコードの公開」事案などについてどう思うか、聞いてみて、事前にその反応やコメントを把握するだけでも、その人の人となりや考え方が見えてくることでしょう。

⑤ 管理体制のメンバーの決定と把握

これまで述べた①～③の維持に加えて、今後ソフトウェアの脆弱性が生じた場合に、組織として対応できるように、管理体制の責任者、外部との連携窓口（Point of Contact）を明確にしておきましょう。

⑥ ステークホルダー（利害関係者）全体の把握

これまで①～③で述べた契約やプロジェクトなどにおいて、様々な関係者が関係してソフトウェアを作っています。その利害関係者をリスト化し、把握しておくようにしましょう。またこのステークホルダーにはソフトウェア開発や維持に関係する外部機関（JPCERT/CC や Software ISAC など）や、場合によっては法執行機関も含めておくようにしましょう。

⑦ ツールやシートを使った管理

これまで述べた①～④の情報を管理するツール（またはエクセルなどのシート）を使って情報を書き出しておきましょう。組織が最低限、維持・管理していくために必要な情報を整理することができます。

## ４．ソフトウェア脆弱性の把握と管理

ソフトウェアは元々の思想からも完全なものを作ることは難しく、何かしらの不備や想定していなかったミスなどが生じます。それが脆弱性です。項番３で述べたような状況把握ができていれば、ソフトウェアの脆弱性に対してもあわてる必要はありません。

脆弱性の対応にそれぞれフェーズがあります。

① 脆弱性の発見

まずは脆弱性を発見する必要があります。発見するきっかけは、OSS の継続的な確認やアップデート、自組織での脆弱性検査、外部からの通報など様々ですが、最も望ましいのは組織内での発見です。また、広く一般的に使われている OSS などであれば、国内外の脅威情報を収集し、整理し、外部発信してくれる組織が情報を公開してくれます。またそのような機関や利用者などから、直接連絡が来る場合も考えられます。適切に連絡を受けるためにも PoC（Point of Contact, 信頼できる窓口）の明確化はとても大切です。いずれにしても、整理してある自社の情報と照らし合わせて、対応するようにしましょう。

なお、脆弱性の対応については、情報処理推進機構において「情報セキュリティ早期警戒パートナーシップガイドライン」が整備されています。このガイドラインを参照しながら、冷静、かつ、迅速に対処するようにしましょう。

検証により、脆弱性を発見する場合は、OWASP が発行し Software ISAC が翻訳した「アプリケーションセキュリティ検証標準」などを参考に検証するレベルや範囲を明確にして検証を実施しましょう。

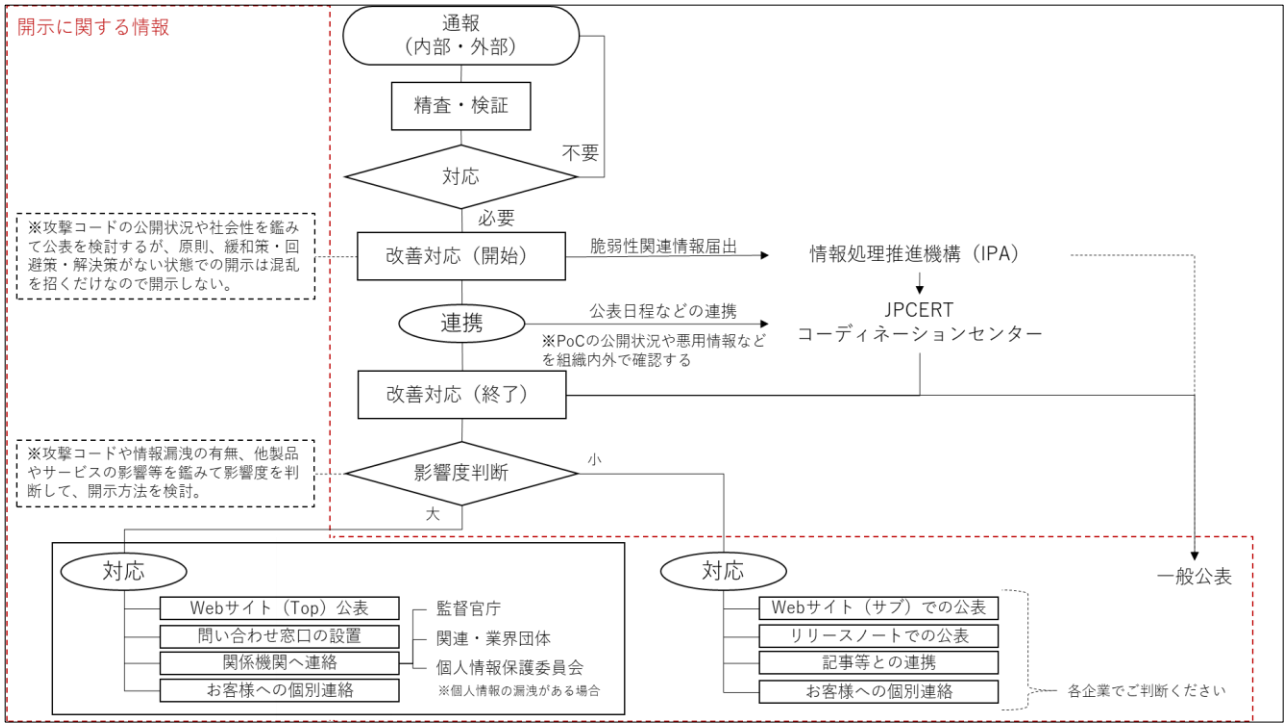


図４　脆弱性対応フロー例

② 脆弱性の判断と改善（発見、判断（トリアージ)、改善、開示）

ソフトウェアの脆弱性に関する確認ができたら、組織としてその脆弱性への対応を検討する必要があります。それは契約や費用、他の回避策や緩和策の有無などから判断をしていく必要があります。当然ながらいくら脆弱性があったからと言って、その組織の事業継続に資するような費用が掛かる場合は対応することができません。項番２で把握した現状と改善に係る費用や改善しなかったときの影響などを鑑みながら、脆弱性の対応をプロジェクトやエンジニアだけではなく、経営者が責任をもって判断をするようにしましょう。特に外部からの指摘の場合は、報告者とコミュニケーションを行い、適切なタイミングで情報開示が行えるよう統制することも大切です。

③ 脆弱性の情報開示

ソフトウェアの脆弱性の改善を行い（または行っている過程において)、情報を外部に発信する必要があります。ここで忘れてはならないのはその脆弱性や対象のソフトウェアの影響が外部に公開されているか否か、そして脆弱性を悪用する攻撃が確認されているかどうかです。もし、あまりにも早く情報を具体的に出すと、自らサイバー攻撃を行う攻撃者に脆弱性を教えることになり、悪用されてしまう場合も考えられます。また OSS の脆弱性で広く情報が公開されている場合で、自組織に関係がある場合には、その時の影響（特にすでに被害が発生している場合など）は、「調査中」である旨を公表し、改善をされ次第、正式に情報を出すことが望ましいでしょう。開示で大切なことは適切にバージョンアップを促すことです。どのバージョンに更新すれば改善されるのか、そのバージョンアップはどのように行うのかなど、情報を開示して利用者への対応を促しましょう。

## ５．普段からのトレーニングと教育

日進月歩で変化するこの時代において、情報と知識のアップデートも欠かすことができません。普段の業務に追われてしまうことも理解はできますが、普段からのトレーニングや教育を疎かにしていると、自組織で開発するソフトウェアの古さや品質そのものに影響がある場合があります。できる限り、外部の研修を受講したり、最新の本や Web などで情報を収集したり、常に新しい情報を追いかけるように努めましょう。また経営者もこのような機会を積極的に承認するようにしましょう。

なお、教育は開発するエンジニアに限った話ではなく、ソフトウェアは組織の事業や経営、広くは日本の産業に大きな影響をすでに及ぼし始めています。そこで最前線に立つ営業や経営者自身もソフトウェアの知識を高めておくことに損はないでしょう。

それでも教育やトレーニングを受けている時間はないという方も少なくはないでしょう。そのような場合はまずは情報収集できるように、IPA や JPCERT/CC などのメルマガに登録して情報を収集するだけでも効果があります。さらには外部連携の活動が業務で行えないとしても、Software ISAC に PoC の情報だけでも登録しておくと、脆弱性情報や教育、そして何よりインシデント発生時などに連携する体制が構築できるので、組織にとって大きなメリットにもなることでしょう。

## ６．最後に

まずは昨今の脅威などを受けてソフトウェア管理体制構築のためのガイド（簡易版）を取りまとめ公開しました。今後、Software ISAC ではガイドの詳細版を作成していく予定です。

情報技術の活用、革新により、時代は大きく変化しています。それを支えているのはソフトウェアであり、今後より一層影響は大きくなることでしょう。言わば私たちはソフトウェアなしには生きていくことができない時代になっています。だからこそ、開発する人や組織、またそれに関係する利害関係者はその責任を理解した上で、「ゆりかごから墓場」まで考えていく必要があります。

また一人で脅威や脆弱性に立ち向かう時代ではなくなってきています。ISAC をはじめとした横の連携と協力によって解決していく時代です。ソフトウェア管理体制と聞くとハードルが高く思うかもしれませんが、まずはこのガイドでソフトウェアを開発する組織が、一歩、また一歩と踏み出して頂けたのならうれしく思います。

ソフトウェア開発を行うステークホルダーで、この国を、そして世界を、支えていきましょう。

別紙

【ソフトウェア開発において参考にすべき情報】

- 一般社団法人コンピュータソフトウェア協会「ソフトウェア出荷判定基準」
  https://www.csaj.jp/NEWS/pr/201201_sec-release-decision.html
- 一般社団法人コンピュータソフトウェア協会「プロダクト脆弱性対策・対応成熟度評価シート」
  https://www.csaj.jp/NEWS/committee/security/200417.html
- Open Web Application Security Project「アプリケーションセキュリティ検証標準」
  https://owasp.org/www-pdf-archive/OWASP_Application_Security_Verification_Standard_4.0-en.pdf
  ※Software ISAC にて上記文書を邦訳
  https://www.csaj.jp/documents/NEWS/pr/20200903_OWASP_ASVS4.0-ja.pdf
- Forum of Incident Response and Security Teams「PSIRT Services Framework」
  (https://www.first.org/standards/frameworks/psirts/FIRST_PSIRT_Services_Framework_v1.0_jp.pdf)
  ※Software ISAC にて上記邦訳を実施
  https://www.csaj.jp/NEWS/pr/191108_psirt.html
- NTIA「Software Component Transparency」
  (https://www.ntia.doc.gov/SoftwareTransparency)
- IPA「情報システム等の脆弱性情報の取扱いに関する研究会」
  (https://www.ipa.go.jp/security/fy2020/reports/vuln_handling/research_2020.html)

【作成者一覧】

- 明尾 洋一（株式会社サイボウズ）
- 襟川 芽衣（株式会社コーエーテクモホールディングス）
- 大三川 彰彦（トレンドマイクロ株式会社）
- 垣内 由梨香（マイクロソフト株式会社）
- 加藤 智巳（株式会社ラック）
- 萩原 健太（グローバルセキュリティエキスパート株式会社／インターバルリンク株式会社）
- 板東 直樹（アップデートテクノロジー株式会社）
- 丸山 満彦（PwC コンサルティング合同会社）