テキスト

自動的に生成された説明

OSS教育資料　(サプライチェーンリスクマネジメント･バージョン)　ライセンス応用

この資料は、各企業がOSSのサプライチェーンリスクマネジメントを行う際に、ライセンスに関する実務（ライセンスの調査、義務の履行、SBOM）やセキュリティリスク対応を習得する上での参考教材です。各社で教育コンテンツを作成する際の材料として自由に複製、改変してご活用下さい。

　本資料は、 [Creative Commons CC0 1.0 Universal](http://creativecommons.org/publicdomain/zero/1.0/legalcode) ライセンスの下でリリースされており、複製、改変、  
配布にあたっての制限はありません。

　　【作成元：[OpenChain Japan Work Group education sg](https://openchain-project.github.io/OpenChain-JWG/subgroups/education/)】

Disclaimer（免責事項）

* 本資料は、日本国内でOSSを活用する場合を前提としており、 education sgのメンバの経験に基づいて説明を記載しています。厳密な法解釈や海外での利用など、判断に迷う場合は専門家にご相談ください。
* 本資料に記載した内容については、作成者、提供元は一切の責任を負いませんので、ご承知のうえご利用ください。
* 本資料は、 OpenChainの仕様書準拠や認定取得を保証するものではありません。

コンテンツ

１．OSSライセンスの調査

２．ライセンス義務の履行

３．SBOM

４．OSSのセキュリティリスクへの対応

1. OSSライセンスの調査  
   1. 導入Q&A  
      ちょっと考えてみましょう。自社でOSS Aを利用しています。インターネット上のWebサイトの記載は、『 OSS Aのライセンスはライセンス　X』となっています。そのOSSのVersion 2を配布する場合、ライセンス　Xで配布して良いでしょうか？答えは、必ず利用するVersion 2のライセンスを確認して、対応するライセンスで配布する必要があります。  
        
      

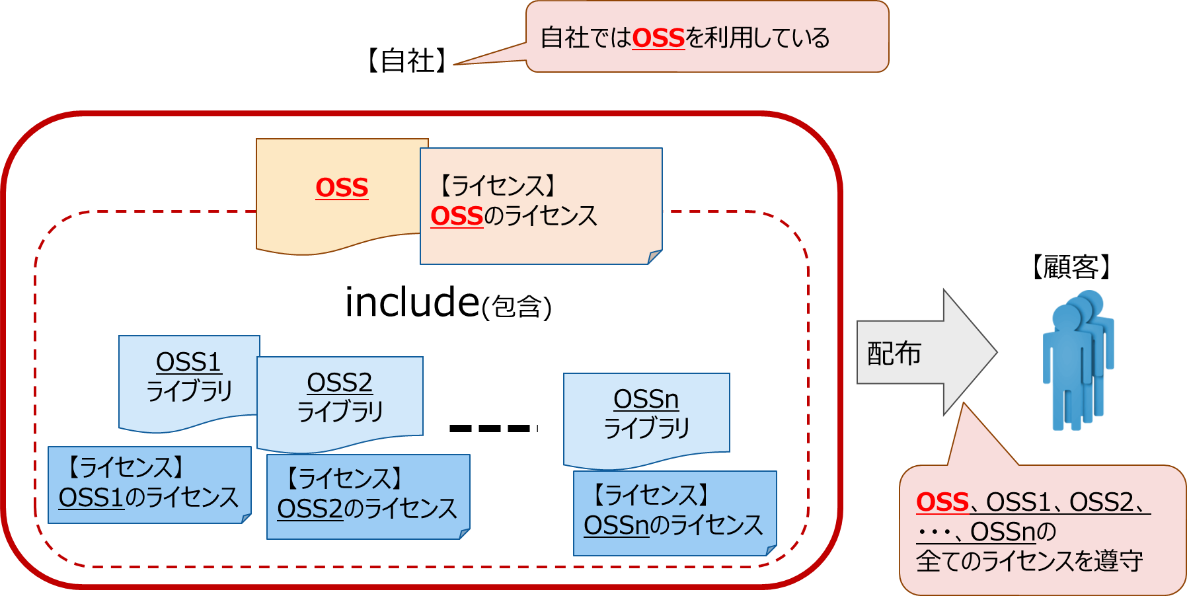
下記の節で、ライセンスの調査に関する事項について確認していきましょう。

* 1. ライセンスの調査  
     　OSSを利用する際には、ライセンスの調査を行い、利用時にライセンスを遵守する必要があります。OSSの選定時には、先ずダウンロードファイルの調査を行い、そのOSSのライセンスを網羅的に調査する必要があります。  
     　ライセンスの調査について、下記を実例として参考に紹介します。
     1. 先ず、対象となるOSSをダウンロードします。  
        ダウンロードする前に、「ダウンロードする時や利用するときはこのライセンスに従うように」等と記載がある場合は、そのライセンスに従う必要があるので、記載事項をライセンス情報として確認します。
     2. (a)でダウンロードしたファイルがアーカイブされたファイル(拡張子が　zip/jar/tar/tar.gz/tgz/xz/deb/rpm等 のファイル)である場合は、これを解凍します。
     3. (a)でダウンロードしたファイルが実行可能形式や Windows Installer 形式のファイル(拡張子が exe,msi のファイル)である場合は、これをインストールします。  
        インストーラを起動し、表示されるライセンス本文やソフトウェア利用許諾契約等をライセンス情報として確認します。
     4. ファイルの名称に下記の文字列が含まれるファイルを検索します。検索されたファイルに　ライセンス情報の記載があるかを確認します。  
        　license/copying/copyright/eula/about.html/pom/manifest-readme/notice/legal/licence(※)等
     5. ダウンロードファイルの全てのファイルの中身に対して、下記の文字列が含まれているかを検索します。これらの文字列が含まれているファイルに、ライセンス情報の記載があるかを確認します。  
        　license/epl/gpl/asl/artistic/legal/notice/third/licence(※)等
     6. (b)でアーカイブされたファイルの場合は、アーカイブされたファイル中の META INF フォルダ内の MANIFEST.MF/pom.xml 等の定義ファイルに、ライセンス情報が含まれていないかを確認します。
     7. ダウンロードサイトと公式サイト、プロジェクトサイトやベンダサイト、が別の場合には、下記を確認します。  
        公式サイト、ベンダサイト上のLicense/Licensing/Readme/Copying/Copyright/Eula/Legal/about.html/notice/Thirdparty/Thirdparties/Licence(※)  
        等の文字列を探し、ライセンス本文やライセンスへのリンクを調査します。

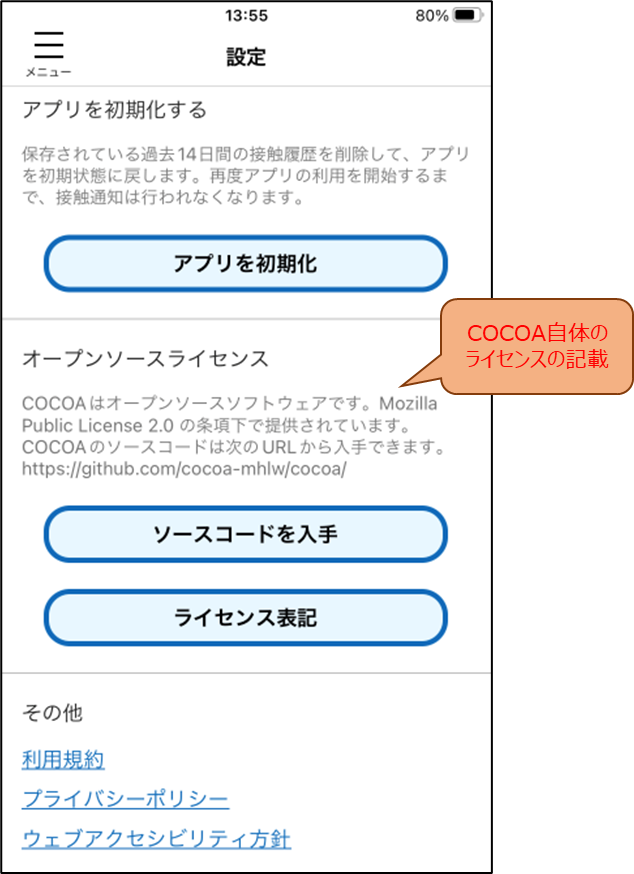
　上記の実施例の(a)～(f)の作業で、ソースコードファイル中のライセンス情報を網羅的に確認する事が基本となります。(g)の作業は、あくまでも補助的な作業と考えた方がよいと思います。(a)～(f)の作業で調査したライセンスと、(g)の作業で調査したライセンスが異なる場合、前者を使用した方がよいと思われます。

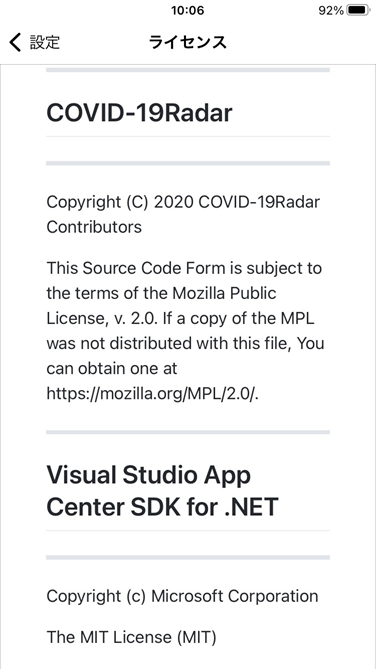
　OSSのオブジェクトコードやロードモジュールを利用する場合でも、対応するソースコードをダウンロードして、ライセンスの調査を行う事が重要です。

　上記の調査でライセンス情報が見つからない場合には、コミュニティの開発者に確認が必要でしょう。それでもライセンス情報が明確にならない場合は、そのOSSは使用・利用禁止にする事が必要になるかもしれません。  
(※: licence,Licenceはlicense,Licenseと同様の意味で使用される場合があります。)

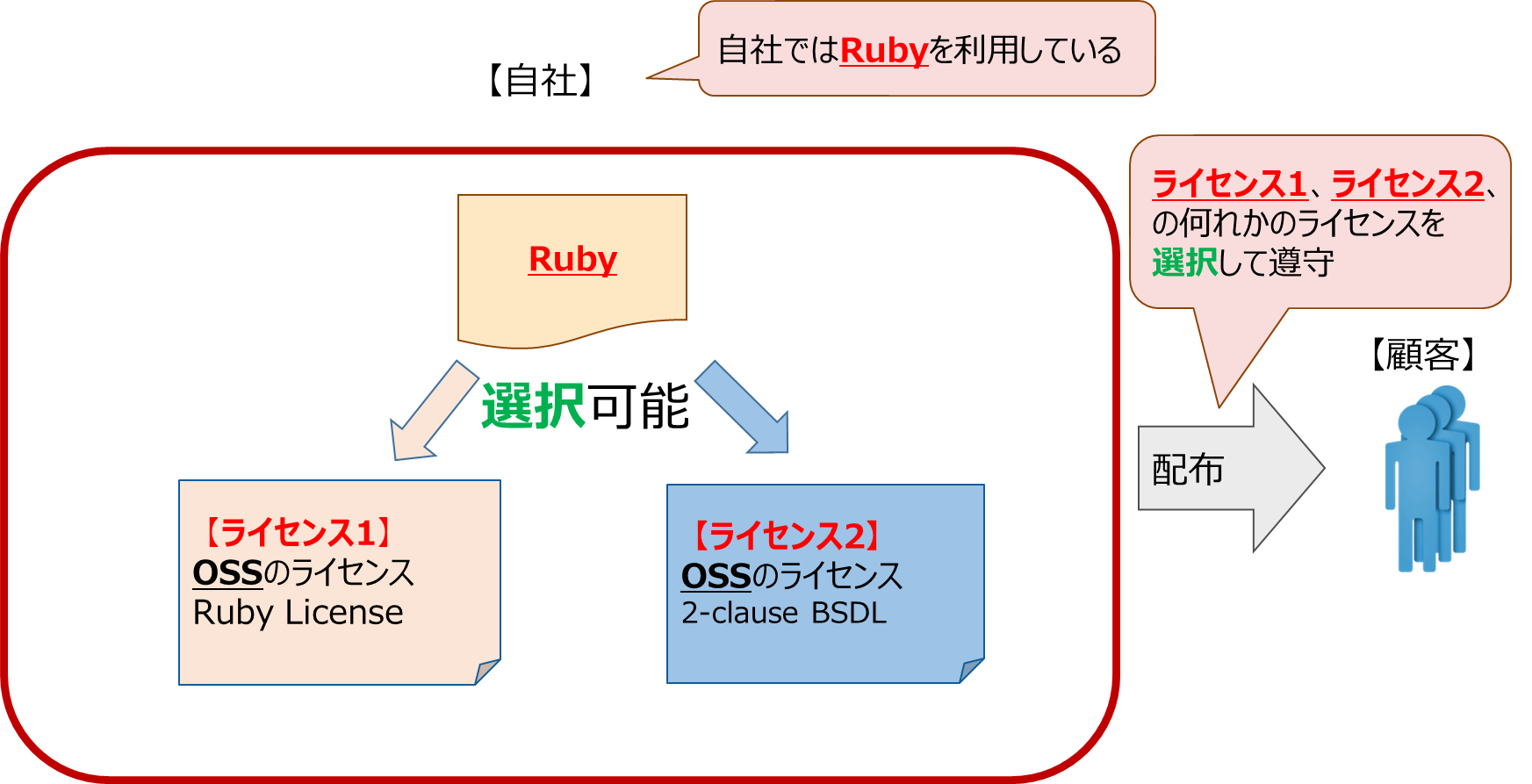
* 1. ライブラリ等のライセンスも確認  
     　近年、OSSも(特にOSやフレームワーク等で)大規模化が進んでおり、他のOSSのライブラリ等を含んでいるものが多々あります。そこで、1.2　ライセンスの調査を確実に行い、網羅的にそのOSSに含まれている他のOSSのライブラリ等とそれらのライセンスを調査する必要があります。  
     

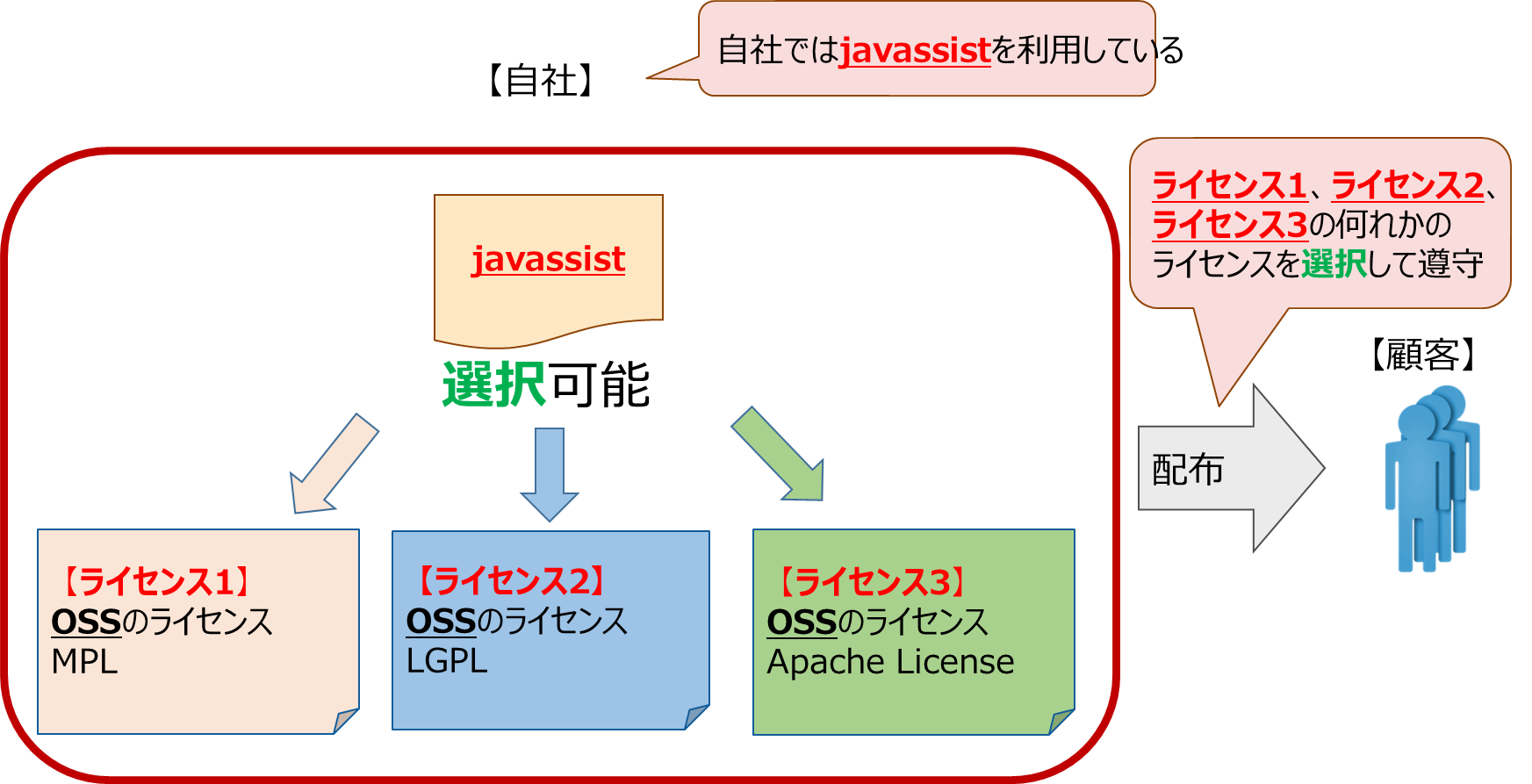
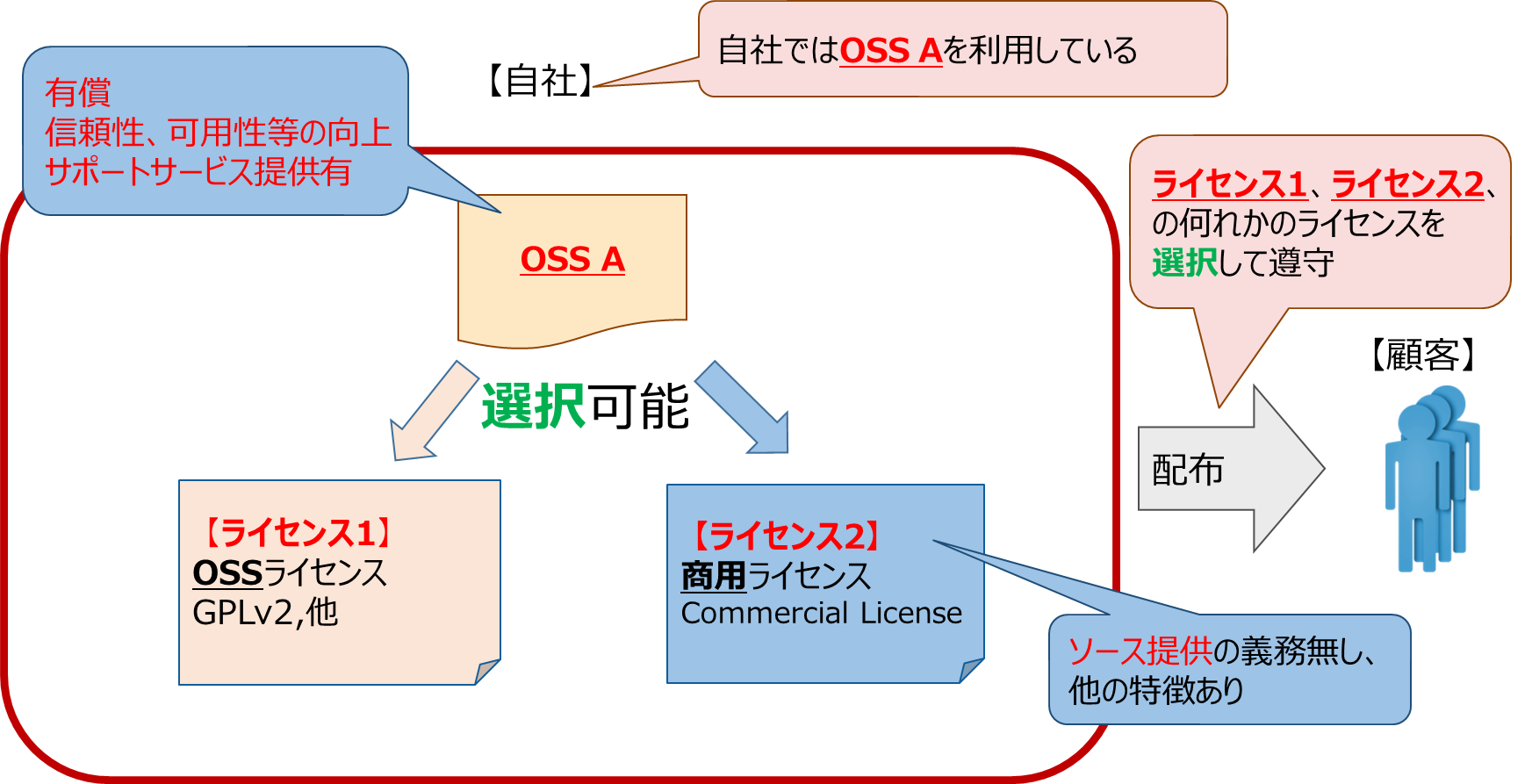
コンテナイメージでの利用を行う場合にも、コンテナの中に含まれるOSS、OSSのライブラリ等とそれらのライセンスを調査する必要があります。

* 1. ライセンスの記載例
     1. OSSをオブジェクトコード、ロードモジュールで利用する場合には、例えば配布するアプリケーションやプログラム上に、ライセンスの記載を行う場合があります。  
        下記は、COCOAのアプリケーションの例を示しています。  
          
        
     2. OSSをソースコードで利用する場合は、ソースコードのファイル群の中にライセンスが記載されていることが場合があります。

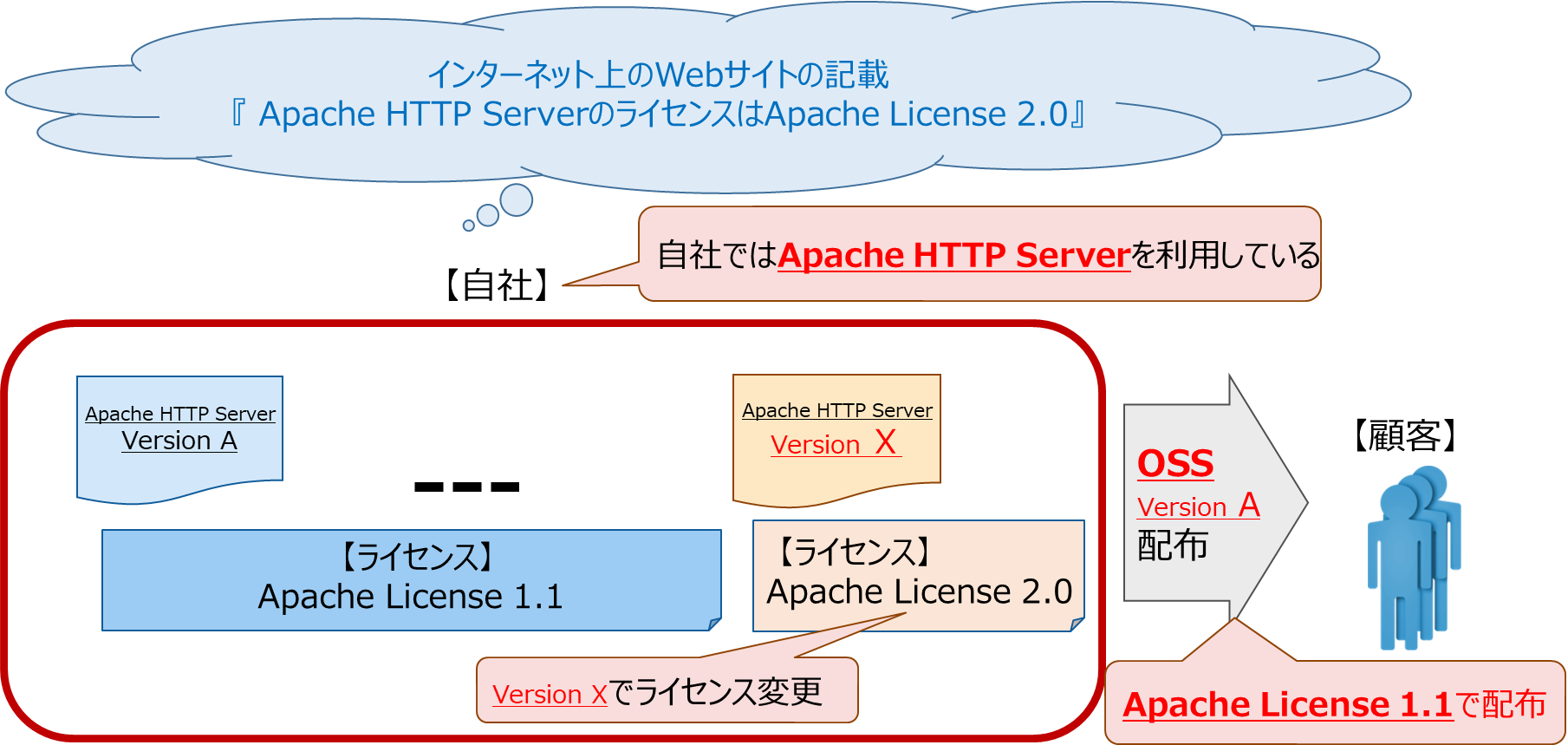
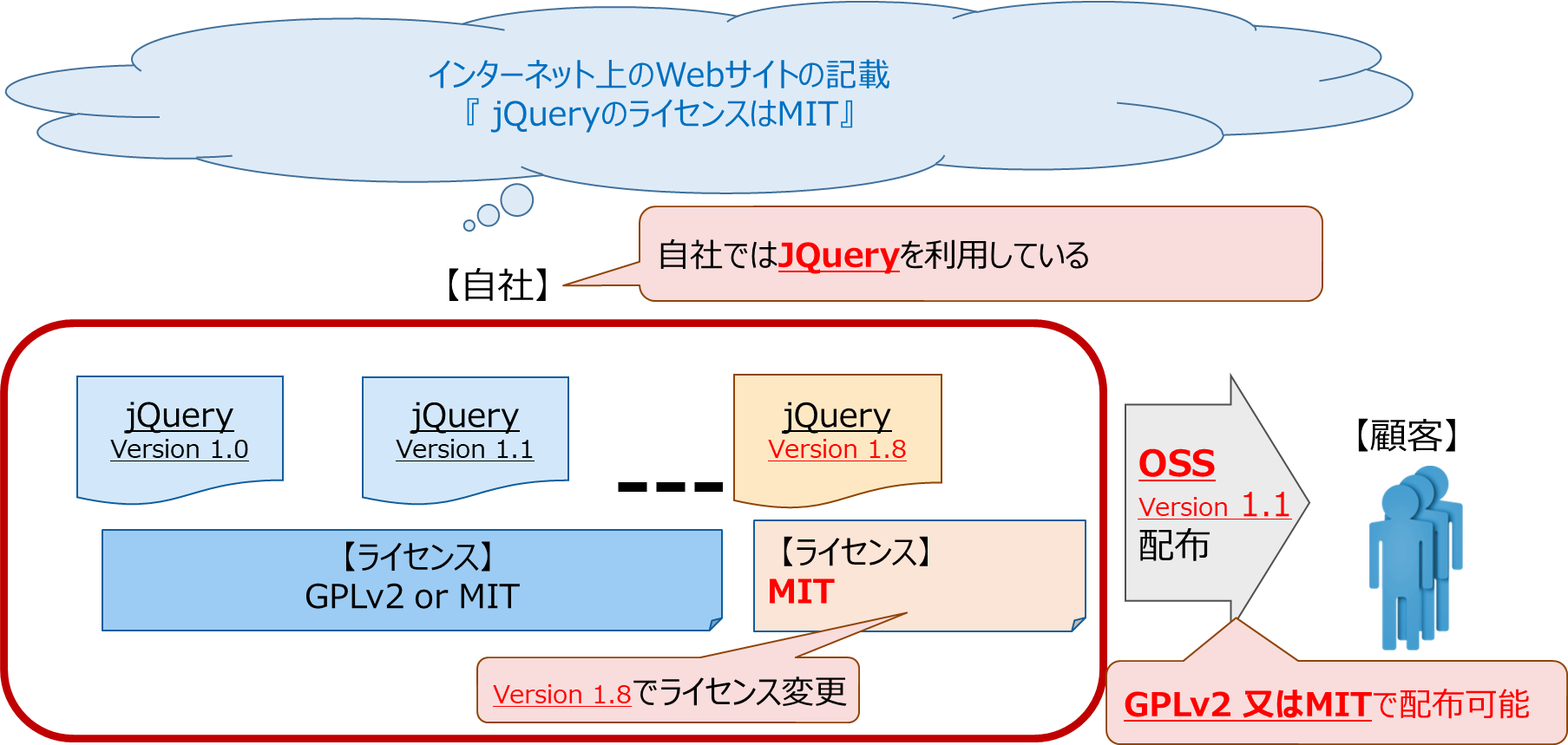


* 1. 複数のライセンスが記載されているケース  
     　OSSの中には、複数のライセンスの中から所望のライセンスを選択して利用できるものがあります。２つのライセンスから選択可能なものはデュアルライセンス、３つのライセンスから選択可能なものはトリプルライセンス、・・・またはマルチライセンスと呼ばれています。
     1. デュアルライセンスの例としてはRubyがあります。下記URLを参照ください。  
        <https://www.ruby-lang.org/en/about/license.txt>



* + 1. トリプルライセンスの例としてはjavassistがあります。下記URLを参照ください。  
       <http://www.javassist.org/>  
         
       
    2. ライセンスの選択については、商用ライセンスとOSSライセンスから選択するケースがあります。ソフト開発ベンダーが、OSSのコミュニティで開発されているOSSに対して、信頼性、可用性等を向上する目的で機能追加し、商用ライセンスで提供するケースになります。一般的に有償で提供され、ソフト開発ベンダーによるサポートサービスが提供される場合があります。また、ソース提供義務のあるOSSライセンスと、ソース提供の義務の無い商用ライセンスを選択できるようになっている場合があります。  
         
       
  1. Web掲載のライセンスで誤認(バージョン)

OSSは、そのバージョンによって、ライセンスが変わっている場合があります。インターネット上のWebサイトには最新のバージョンのライセンスが記載されている事が多く、バージョンの古いOSSにも同じライセンスが適用されると判断してしまうと、OSSを配布する場合に、誤った異なるライセンスで配布してしまう恐れがあります。必ず、配布するバージョンのライセンスを確認しましょう。  
  


* + 1. 事例１としては、Apache HTTP Serverがあります。Apache HTTP Serverは、Version Xになる際に、Apache Licenseも1.1⇒2.0となりました。  
         
       
    2. 事例2としては、jQueryがあります。jQueryは、Version 1.8より以前は、GPLv2とMITのデュアルライセンスでしたが、Version 1.8でMITに統一されました。  
         
       
  1. OSS、ライセンスのリスト化
     1. 1.2～1.6の様にOSSのライセンスを調査して確認する事は、OSSを利用する際にそのライセンスの義務を履行するためにも重要になります。又、第３章のSBOMを作成するためにも必要になります。  
        そこでライセンス調査の結果は、下記の様な項目についてリストにして纏めておくと良いでしょう。  
        (1)OSS名称  
        (2)OSSのバージョン  
        (3)OSSのダウンロードURL  
        (4)ライセンス名称  
        (5)著作権等の表示
     2. OSS、ライセンスのリスト化を行っておくと、社内でOSSのコンプライアンスについて検討、承認を行う際に、大変有効に活用できます。

1. ライセンス義務の履行
   1. 導入Q&A  
      　ここでひとつ例を考えてみましょう。開発効率向上のためにOSS Aを製品に搭載することになりました。OSS Aはその分野では非常に有名であるため、その名前や開発者については誰でも知っています。

このとき、たとえば製品のどこかに「OSS Aを利用しています」と一言添えれば、OSSライセンスの遵守作業としては十分でしょうか。

残念ながら話はそう単純ではなく、OSS Aのライセンス条件によっては、十分に遵守された状態とは言えません。著作権侵害にならないためにはそのOSSライセンスをきちんと理解し、その条件を遵守する必要があります。本節ではOSSライセンスの代表的な遵守作業と、そのポイントについて確認していきましょう。

* 1. ライセンス遵守作業の概要  
     　下記はOSSライセンスの遵守作業の中でも、特に代表的なものです。  
     + ソースコードを提供する
       - レベル３（※）のOSSライセンスでは、OSSそれ自体のソースコードを提供します。このときOSSに対する改変有無は問いません。
       - レベル4（※）のOSSライセンスでは、レベル３の対応に加え、さらにOSSの結合著作物とみなされる他のプログラムのソースコードも提供します。

Table X.　ソースコードの提供要否

|  |  |  |
| --- | --- | --- |
| OSSライセンスのレベル（※） | OSSの ソースコード | OSSの結合著作物とみなされる自社開発のソースコード |
| レベル１（ライセンス文の添付） | 不要 | 不要 |
| レベル２（謝辞や著作権情報の追加） | 不要 | 不要 |
| レベル３（OSSのソースコード提供） | 提供が必要 | 不要 |
| レベル４（連携プログラムのソースコード提供） | 提供が必要 | 提供が必要 |
| レベル５（サービス時のソースコード提供） | 提供が必要 | 提供が必要 |

※ここでの「OSSライセンスのレベル」とは、ライセンス条件の強さを5段階に類型化することで、リスク評価の簡易化を図ったものです。各レベルの詳細については基礎編の第2章をご参照ください。

* + - リバースエンジニアリングを許可する
      * レベル４の一部のOSSライセンスで求められる条件です。通常、製品の多くはリバースエンジニアリングを禁止していますが、ここではOSSとリンクしたプログラムについてリバースエンジニアリングを禁止しないよう製品のドキュメントを修正します。
    - 製品のドキュメントに、必須文言（著作権表示他）を掲載する
      * 大半のOSSライセンスで求められる条件です。著作権情報やライセンス条文、免責条項などを製品のドキュメントに掲載し、受領者にOSSに関する基本的な情報を開示します。
  1. ソースコード提供  
     　OSSをバイナリコードの形式で製品に含める場合、受領者がそのバイナリコードを正確に再現するために必要な情報を提供するよう求めるライセンスがあります。
* レベル３のOSSライセンスではOSSそれ自体のソースコードを提供します。OSSに対する改変有無は問いません。
* レベル4のOSSライセンスでは、レベル３の対応に加えて、OSSの結合著作物とみなされる自社開発部分のソースコードも提供します。
  1. 提供するソースコードの準備  
     　製品に含めたOSSのバイナリコードを再現するために必要となる全ての情報を用意します。具体的には①対象のソースコードと、②その再現に必要な情報の2点です。
* バイナリコードが過不足なく再現できるように、ソースコードに加えて、バイナリコード生成時の情報も用意します。コンパイルオプションや各種設定パラメータ (一般にmakeファイル一式)がこれに相当します。なおコンパイラなどの「バイナリコードを再現するための環境」それ自体を提供する必要はありません。
* OSSを改変した場合はその箇所を告示する等、一般的なコーディングルールやマナーに配慮するとよいでしょう。
* たとえばソースコードを敢えて紙媒体で配布する等、バイナリコードの再現を意図的に妨げる行為は避けましょう。同様に、たとえばOSSにもともと含まれている他人の著作権表示を自社のものに差し替える等、著作権を侵害する行為も避けましょう。
* ソースコードの提供窓口はOSSの開発コミュニティではなく、バイナリコードの作成者である製品開発元が担います。仮に受領者に対して、製品開発元の管理が及ばないWebサイトをソースコードの提供窓口として案内してしまった場合、その後の意図しないサイトの閉鎖や移転によって「ソースコードの提供」というライセンス条件を満たせなくなってしまう可能性があるためです。提供対象のソースコードは製品開発元できちんと保持し、OSSの受領者に適切に提供できるよう、体制を整えておきましょう。

タイムライン が含まれている画像

自動的に生成された説明

Fig X. ソースコードの提供者はバイナリコードの作成者

* 1. 提供方法の選択  
     　OSSライセンスの中には、ソースコードの提供方法として複数の選択肢を提示しているものがあります。そのようなケースでは自身の状況に最も適した提供方法を選び、その方法にしたがってソースコードを提供すると良いでしょう。たとえばレベル４の代表的なOSSライセンスであるGNU General Public License (GPL)の場合、ソースコードの提供方法は下記の３点から選択可能です。
* 製品に含めるOSSのバイナリコードとともに、その再現に必要なソースコード一式も同梱する
* 少なくとも3年間有効な、ソースコード一式を提供する旨の文書を同梱する
* ピア・ツー・ピア伝送を使う（ただしバージョン3に限る）

これらはソースコードおよびその案内文書の同梱の仕方により、さらに細かく分かれます。事業や製品の特性、OSSライセンスに応じた適切な方法を選択することが重要です。

1. 製品に含めるOSSのバイナリコードとともに、その再現に必要なソースコード一式も同梱する  
   　両者を製品本体または製品のドキュメント（紙媒体や電子媒体）に直接同梱する方法です。  
   　製品出荷時にバイナリコードとソースコードを同梱するため、その時点の正確な情報が確実に同梱されるという利点があります。また後述の「ソースコードを提供する旨の案内」と比較して、ソースコードを提供するための体制の構築や維持も不要になります。  
   　一方で注意したいのは、製品本体のバージョンアップの際はバイナリコードと一致した正しいソースコードもきちんと同梱されているか適宜見直す必要がある点や、製品本体や製品のドキュメントの容量を圧迫する点などです。バージョンアップの頻度や製品のドキュメント形態によって最適な方法は異なるので、それぞれの事情に合った方法を選択しましょう。

ダイアグラム

自動的に生成された説明

Fig X. ソースコードを同梱する三つの形態

1. 少なくとも3年間有効な、ソースコード一式を提供する旨の文書を同梱する  
   　「受領者のリクエストに応じてソースコードを提供する」旨の案内を、製品本体または製品ドキュメント（紙媒体や電子媒体）に同梱する方法です。  
   　案内に基づき受領者が製品開発元に対して「製品搭載されているOSSのソースコードを提供してほしい」と依頼し、それに応じる形でソースコードを提供します。この方法を採用した場合、製品開発元はさらに下記の手段を講じることができます。  
   * 1. ソースコードを提供しているWebサイトのURLを案内に含める
     2. ソースコードを格納した媒体を提供する窓口を案内に含める

案内の掲載場所は、受領者が製品を使用する過程で自然に目を通すものであれば、紙媒体でも電子媒体でもかまいません。製品に付帯する紙の取扱説明書やエンドユーザライセンス（End-User License Agreement, EULA）があればそこに記載してもよいですし、製品の特性として何らかの表示機能を有していればオンラインヘルプでも問題ありません。

ダイアグラム が含まれている画像

自動的に生成された説明

Fig X. 少なくとも3年間有効な、ソースコード一式を提供する旨の文書を同梱する

* ソースコードを格納した媒体（例：CD-ROMやDVDなど）を提供する際は、送付に必要な実費を請求することも可能です。その場合は郵送費や媒体費を鑑み、妥当な価格を設定しましょう。
* 「ソースコードを提供する」旨の案内を行う際は、その意思が明確に表現されているか確認しましょう。

GNU General Public License version 3.0 (GPL v3.0)の場合、前記２点の方法のほかに、ピア・ツー・ピア伝送（peer-to-peer transmission）を利用した方法もあります。

* 1. 提供の実施

下記は自身の管理するWebサイト上でソースコードを提供した例です。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

Fig X. 新型コロナウィルス接触確認アプリ「COCOA」のソースリポジトリでは、  
COCOAが利用しているOSS「Project Covid19Radar」のソースコードを提供している  
（https://github.com/cocoa-mhlw/cocoa）

また、下記はしばしば陥りがちな注意点です。

1. 製品に含めるOSSのバイナリコードとともに、その再現に必要なソースコード一式も同梱するケース  
   正しいソースコードが適切に同梱されているか、製品の最終的な出荷物件を確認しましょう。  
   たとえば製品出荷に至るまでの一連の工程でも、最新版の​​​​​​​ソースコードを開発部門が下流工程の部門へ提供できていないケースや、出荷部門の判断でソースコードが出荷対象から除外されてしまうケースなどが考えられます。
2. 少なくとも3年間有効な、ソースコード一式を提供する旨の文書を同梱するケース  
   ソースコードの受領者にとってわかりやすい案内となっているか、確認しましょう。  
   1. ソースコードを提供しているWebサイトのURLを案内に含める場合
      * バイナリコードを再現可能なソースコードを、フォルダ構造を維持したまま提供しましょう。
        + 「バイナリコードの再現を意図的に妨げている」と捉えられないように、過度のファイル分割や公開先の分散は避けましょう。
        + Webブラウザ上でソースコードをそのまま確認できる状態になっている必要はありません。

makeコマンド等で一括再現可能であればtgz形式やzip形式などの圧縮形式で提供してもかまいません。

* + - 製品に含まれるOSSを変更した際は、Webサイトで提供しているソースコードも忘れずに更新しましょう。
      * OSSを最新版に差し替えた、あるいは改変を加えた場合は、製品に含まれるOSSと一致させるためにWeb上で提供しているソースコードも適宜差し替えましょう。
      * 一方、最新バージョンだけではなく旧バージョンの製品も市場に残るような場合、最新バージョンと同様に旧バージョンのソースコードも引き続きWeb上で提供される必要があります。
  1. ソースコードを何らかの媒体で提供するための窓口を案内に含める場合
     + 製品のバージョンごとにソースコードを用意しましょう。
       - 提供するソースコードは、受領者の持つ製品に含まれたバイナリコードに一致している必要があります。正しいソースコードを提供するためには、製品の構成管理が重要です。
       - 初めから媒体を用意しておく必要はありません。あらかじめ用意しておいても、あるいはソースコードの提供依頼があった際に初めて作成しても問題ありません。
       - makeコマンド等で一括再現可能であればtgz形式やzip形式などの圧縮形式で提供してもかまいません。
     + ユーザサポート部門と事前に手順をすり合わせておきましょう。
       - たとえ社内に窓口を新設する場合であっても、既存の窓口にソースコードの提供依頼が入る可能性があります。適切な窓口にエスカレートされるよう既存の窓口にも手順を確認しておく方とよいでしょう。
       - ユーザサポート部門は適切な回答を行うために、ソースコードの提供を求められる（正当な理由のある）製品と、そうでない製品を常に把握しておく必要があります。

　OSSライセンスが引き続き遵守されるよう、製品出荷後においても適切な状態でソースコードが提供されているか定期的に確認するようにしましょう。

* 1. リバースエンジニアリングの許可  
     　製品を販売する際、購入者に対してエンドユーザライセンスを提示する手法を採ることがあります。

　これは権利の帰属と製品の想定使用範囲をあらかじめ明示しておくことで、双方の意図しない権利侵害や事故、損害を防ぐための一般的な措置ですが、多くのエンドユーザライセンスではここに「リバースエンジニアリングの禁止」も含んでいます。特に昔ながらのエンドユーザライセンスでは「本製品に含まれる知的財産権はすべて弊社に帰属します。」といった表現をしばしば含みますが、これはOSSに限らず第三者によるソフトウェアを数多く利用する現在の製品開発においては実態に合わない記述になりつつあります。

そのため昨今では、第三者のソフトウェアにまで権利を主張しないよう「例外条文」を設けることが一般的です。

例外条文の例１：

THE SOFTWARE MAY INCLUDE SOME PARTS LICENSED TO YOU BASED ON THE LICENSE AGREEMENTS OTHER THAN THIS AGREEMENT AND THE LICENSE AGREEMENTS FOR OUR SOFTWARE PRODUCT. IN SUCH CASE, THE LICENSE AGREEMENTS OR TERMS FOR SUCH PARTS SHALL TAKE PRECEDENCE OVER THIS AGREEMENT AND THE LICENSE AGREEMENTS FOR OUR SOFTWARE PRODUCTS.

（参考訳）許諾ソフトウェアには本使用条件以外のライセンス契約に基づきお客様の使用許諾される部分が含まれることがあります。この場合、かかる部分に関してのご使用の条件は当該ライセンス契約以外の条件が本使用条件よりも優先します。

例外条文の例２：

許諾ソフトウェアの中には、B社以外のソフトウェアの権利者が定める使用許諾条件（GNU General Public license (GPL)、Lesser/Library General Public License (LGPL)を含みますが、これらに限られるものではありません）を伴うソフトウェア（以下「対象外ソフトウェア」といいます）が含まれている場合があります。対象外ソフトウェアの使用は、各権利者の定める使用許諾条件に従っていただくものとします。

対象外ソフトウェアには、①ソースコードの形式でまたは無償で公に入手可能なソフトウェアを含むものまたはその派生物であり、かつ②本契約の規定と異なる定めの適用を受けるソフトウェア（対象となるソフトウェアおよびその派生物をソースコードの形式で開示または頒布する義務、対象となるソフトウェアを任意の第三者に対して自由に使用許諾させる義務等を含みますがこれに限られません。また、これには GNU General Public License (GPL)やGNU Lesser/Library General Public License (LGPL)に基づいてライセンスされているソフトウェアが含まれますがこれに限られません。）（以下「オープンソースソフトウェア」といいます）が含まれることがあります。

B社が開示するオープンソースソフトウェアのソースコードは、 B社の指定するサイトにてご確認下さい。オープンソースソフトウェアには、それぞれのオープンソースソフトウェアに該当するライセンス条件が、本契約の代わりに適用されます。

例外条文を追加する場合はエンドユーザライセンスの担当部門と連携し、販売形態も含めた製品特性に合った方法を検討しましょう。エンドユーザライセンスに「ひな形」があればそちらを修正流用することも良いですが、最終的なエンドユーザライセンスの確認はかならず行うようにしましょう。

* 1. 必須文言（著作権表示他）を製品のドキュメントに掲載する  
     　大半のOSSライセンスでは、その著作権情報やライセンス条文、免責条項など、OSSに関する基本的な情報も受領者に開示するよう求めています。開示内容はOSSライセンスによって異なり、ライセンス条文を製品のドキュメントに掲載するのみで良いものもあれば、ライセンス条文に加えて謝辞の掲載も求めるもの、あるいは広告媒体上での謝辞の掲載を必要とするものもあります。

開示方法の一例は下記のとおりです。より具体的な条件がライセンスに詳述されている場合もあるため、情報開示を検討する際は各ライセンスの条文を実際に確認することが重要です。

* 取扱説明書やエンドユーザライセンスなどの製品のドキュメントに含める
* （製品自体に情報表示機能がある場合）オンラインヘルプやオプションメニューなど、ユーザが使用する機能の中で表示する
* （HTML形式など、ソースコードのまま第三者に提供する場合）ソースコードに予め埋め込まれた著作権表示を削除せずに残しておく

注意点として、開示は「OSSの受領者にとって自然に見る場所」で行う必要があります。

例えば、「取扱説明書やエンドユーザライセンスが紙の冊子であるにも関わらず、何の案内も無いままに、OSSに関する著作権情報やライセンス条文のみを専用のCD-ROMで保管する」といった方法は、あまり好ましくありません。OSSの受領者が意図的に「製品に含まれるOSSの情報を調べよう」としない限り、目に触れることがないからです。もし他の製品ドキュメントと分離して開示する場合は、その旨を明確に案内する文書を追加するなど、OSSの受領者にとってわかりやすい形となるよう配慮しましょう。

* 1. 製品のドキュメントにライセンス条文を掲載する  
     　OSSライセンスの条文は多くの場合、ソースコードのルートディレクトリにある「COPYING」や「LICENSE」という名前のテキストファイルに格納されています。ライセンス条文の多くは著作権情報やライセンス条件（与えられる権利と制約事項、遵守すべき条件）、免責条項が英文で記載されており、大抵はこれをそのままOSS受領者に開示することでライセンス条件を満たすことができます。

　下記はレベル２のOSSライセンスの例です。上から順に著作権情報、ライセンス条件、免責条項についてそれぞれ記載されています。

図形

自動的に生成された説明

Fig X. レベル２のOSSライセンス  
（著作権表記の年数と権利者が未記入の状態の、MITライセンスのテンプレート）

* 1. 謝辞を掲載する  
     　レベル２の古いOSSライセンスの中には、ライセンス条件として「謝辞」の掲載を求めるものがあります。この場合、前述のライセンス条文と同様に、OSS受領者に対して下記のような情報開示を行う必要があります。

グラフィカル ユーザー インターフェイス, テキスト

自動的に生成された説明

Fig X. 謝辞の記載例

テキスト, 手紙

自動的に生成された説明

Fig X. 謝辞の掲載を求めるライセンス例１  
（Apache Software License 1.1, https://apache.org/licenses/LICENSE-1.1）

テキスト, 手紙

自動的に生成された説明

Fig X. 謝辞の掲載を求めるライセンス例２  
（Libjpeg License, https://jpegclub.org/reference/libjpeg-license/）

* 1. 謝辞を広告媒体に掲載する  
     　前述同様に、レベル２の一部のOSSライセンスには、「このソフトウェアの機能または使用について言及するすべての広告媒体にも謝辞を掲載」するよう求める条項があります。

テキスト

自動的に生成された説明

Fig X. 謝辞を広告媒体へ掲載するよう要求するライセンス条項の例  
（BSD 4-Clause Original (Old) License）

1. SBOM（忍頂寺）
   1. ソフトウェアの構成を知る

ここまで、オープンソースソフトウェア(OSS) においてライセンスコンプライアンスが重要だということを見てきました。

1. 自分が利用しようとする利用しようとするOSS:
2. そのOSSが開発者以外のOSSを含む場合: OS
3. そのOSSが他のOSSを利用する場合: O

※悩んでしまった…

* 1. SBOMを取得する方法

SBOMがあれば、利用するソフトウェアのライセンスや著作権などの情報を把握や、関連する脆弱性情報と紐づけることができます。SBOMを取得する手段には、次の手段が考えられます。取引先や事業環境に応じて適切に手段を選択することが重要です。

1. ソフトウェアの提供元から入手する：
2. 手作業で解析する：
3. ソフトウェSCA (Software Composition Analysis：ソフトウェア構成解析)のツールやソリューションを利用する：
   1. SBOMで扱う管理項目  
      　　※ ソフトウェアコンポーネントを特定できることとしてminimum elements などを概説か。加えて、

* 1. SBOMのフォーマット

先に挙げたMinimum Elementsでは、SPDX、CycloneDX、SWIDのどれかを採用し、かつ、いずれのフォーマットにも相互に変換できることが必要だとしています。ここでは、オープンスタンダードのSBOMフォーマットについて紹介します。

* + 1. SPDX：
    2. SPDX Lite：SPDXのサブセットとして仕様化されています (ISO/IEC 5962及び SPDX v2.3では Annex.G)。パッケージのレベルで管理することと、Excelなどのスプレッドシートで読み書きできるフォーマットとして利用できることを想定し、SPDXのフルセットよりも取り扱う管理情報を絞ったものにしています。取引によっては、SPDX Liteまたはこれをカスタマイズしたものを利用することがあるでしょう。
    3. CycloneDX：Open Web Application Security Project（OWASP)が
  1. SBOMを管理するためのベストプラクティス
     1. SBOM管理のポリシーとプロセスを定める: ISO/IEC 5230 (OpenChain Specificartion) 、とくに§3.3 “Open source content review and approval” に照らして、SBOMの作成と管理に関して、ポリシーとプロセスを定め、文書化し、エビデンスとして管理するようにしましょう。
     2. ソフトウェアライフサイクルに応じて更新し、管理する:　ソフトウェアの構成に変更がある時、都度、SBOMを更新するようにしましょう。たとえば、ソースコードの状態、ビルド用に準備を整えている状態、ビルドしたバイナリの状態、リリースして運用後にソフトウェア更新をした場合、などで、構成が変わることがあります。SBOMは、利用している状態を反映して最新のものになっていることが重要です。
     3. SCAツール等や、構成管理システム: OSS自体の大規模化や、間接的に依存するOSSが多い場合などがあります。ツールなどを利用して効率よくSBOMを作成しましょう。ソフトウェア開発環境によってはCI/CDと連携させることも検討すると良いでしょう。また、SBOMはいつでも確認できるように、構成管理システムを用意するなど、集約しておくと良いでしょう。
     4. e-ラーニング、トレーニング: ルール、プロセス、ツール等の使い方、エビデンスの管理など、開発者及びSBOM管理の運用者にトレーニングを実施しましょう。SBOM管理が当たり前の状態となることで、コンプライアンスや脆弱性に係る問題発生の予防や、問題発生時の速やかな対処に繋がります。
     5. 法的要請への対応、業界や取引先への対応、SPDX Lite の利用：法的な要請によりSBOM及びそれと関連付けて必要となる管理情報が定まる場合があります。また、業界や取引先によっては、そうした法的要請への対応に加えて独自で必要とする管理情報を設定している場合もあります、できるだけ早めに情報収集し、必要に応じて取引先と協議し、対応を進めるようにしましょう。また、SPDX Lite が扱う情報で十分の場合もあれば、これを変更して利用する場合もあるでしょう。
     6. オープンソースコミュニティへの参加：SBOM管理については、様々なオープンソースコミュニティで情報交換がなされています。SBOMのフォーマットであるSPDXやCycloneDXはオープンスタンダードのため、それぞれのオープンソースコミュニティでその仕様策定や管理が議論されています。また、オープンソースコンプライアンスのためのツールを開発するコミュニティでもそうした活動が見られます。さらに、SBOMの相互利用性を検証するために、SBOMやツールのコミュニティが共同し、Plugfestを開催する場合があります。こうしたコミュニティに参加することで、共通の課題の発見や、コミュニティと共に課題解決に取り組むことができます。
  2. ケーススタディ、事例

経済産業省が、SBOMの運用に関する事例、

* + 1. 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（2022年5月拡充版）  
       <https://www.meti.go.jp/press/2022/05/20220510001/20220510001.html>
    2. 参考資料　SBOM（Software Bill of Materials）の導入に関する手引（案）<https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/software/009.html>
  1. 発展的な話題

※NTIA, CISA, EU CRA, EO14028, 経済安全保障推進法案、などなど？

※SBOM Attestation。OpenSSF におけるSBOM関連の取組。

※各ツールの紹介？

1. OSSのセキュリティリスクへの対応
   1. はじめに

　現代のソフトウェア開発において重要な役割を果たすOSS（オープンソースソフトウェア）には、注意すべきリスクが存在します。2021年には、脆弱性が発見されたログライブラリ「Apache Log4j」が社会的な問題となりました。このようなOSSの脆弱性を悪用した攻撃は増加しており、Node.js (NPM) および Python (PyPI)の[パッケージリポジトリを標的とした攻撃も報告](https://jpn.nec.com/cybersecurity/blog/211217/index.html)されています。こういった背景から、企業や組織は今まで以上のOSSのセキュリティ対策を求められてきています。

OSSには同じような機能を持つものや、同じコンポーネントの異なるバージョンによって、異なるセキュリティレベルが存在することがあります。このため、企業は使用目的に応じた適切なOSSコンポーネントとバージョンを選択する必要があります。また、OSSには多数のコントリビューターが存在し、キュリティ修正も行われていますが、いつ何時でも新たな脆弱性が発見される可能性があるため、最新情報を収集し、適切な対応策を講じる必要があります。OSSの使用に際しては、セキュリティに対する十分な注意が必要です。

本章では、OSSのセキュリティリスクについて分析し、リスクに対する有効な対策と、OSSのセキュリティに関する対策方法も紹介します。また、OSSのリスクに関するレポートや事例、有用なガイドラインも紹介します。想定読者は、開発者やシステム管理者、セキュリティ担当者、OSS推進担当者などです。OSSのセキュリティの重要性を再確認し、リスクに対処するためのヒントにしていただけると幸いです。

* 1. OSSのセキュリティリスクの種類

オープンソースソフトウェア(OSS)多くの人々が開発に携わっていますが、その質は様々です。扱うにあたってどのようなセキュリティのリスクがあるか、考えてみましょう。

1. 脆弱性のあるバージョンの採用：過去にリリースされた古いバージョンのOSSはセキュリティの問題がある可能性があります。これは、新しい脆弱性が発見された場合に、OSSプロジェクトは脆弱性を修正した最新バージョンリリースするためです。古いバージョンのOSSを使用する場合、攻撃者は古いバージョンに脆弱性を見つけ、その脆弱性を悪用して攻撃することができます。さらに、古いバージョンは、古いプロトコルを使用しているため、この点で現在のセキュリティ基準に適合しないことがあります。
2. パッケージ・ライブラリの脆弱性: 使用しようとするOSSには、それらが依存するパッケージやライブラリがある場合があります。依存関係に脆弱性がある場合はa). で挙げたリスクと同様にアプリケーションやシステムに影響を与える可能性があります。
3. パッチの未適用： OSSプロジェクトが脆弱性に対する修正パッチをリリースした場合、Linuxディストリビューションなどのリポジトリ管理者はOSSプロジェクトがリリースした修正パッチをリポジトリへ配信します。システムの管理者は配信された修正パッチを適用しなければ、脆弱性の脅威に晒され続けます。
4. ソースコードが公開されているためのリスク：オープンソースソフトウェアのソースコードが公開されているため、悪意のある人々が脆弱性を探し出し、それを悪用する可能性があることには注意が必要です。しかし、OSSは多数の人々によって監視され、改善されることが多いため、協力的なコミュニティの存在がそのリスクを軽減することができます。この点はメリットとリスクが表裏一体となっています。公開されているコードはユーザーが多いほど脆弱性が早期に発見される可能性が高くなり、また修正される可能性も高くなる傾向があります。
5. 開発プロジェクトのサポート: OSSは、開発プロジェクトによって開発が行われるため、バグや脆弱性の修正対応は開発プロジェクトが行います。ユーザーコミュニティや企業などが脆弱性の対応支援を行う活動もあり、第三者から修正プログラムが提供される場合があります。しかし、活発なコミュニティでない場合や、セキュリティに特化したコミュニティでない場合は、問題の解決が遅れる可能性があります。(\*)

(\*)State of Open Source Security 2022. https://snyk.io/reports/open-source-security/

OSSプロジェクトの脆弱性修正にかかる時間が増加したとの調査結果. <https://www.conversion.co.jp/tecblog/20220805>

OSSはこのような上記したリスクがあるため、プロプライエタリ・ソフトウェアなど非オープンソースソフトウェアと比較されることがありますが、どちらが安全か決めることはできません。どのようなソフトウェアもセキュリティに対するリスクを持ちますが、リスクを正しく評価し、適切な対策を講じることが重要です。

* 1. OSSのセキュリティリスクを発見する方法

既知の脆弱性を発見する方法として、以下の手段があります。これらの方法を組み合わせ、組織やプロジェクトの状況に応じたベストな脆弱性の収集方法を決定しましょう。

1. セキュリティ情報配信サービスの利用: セキュリティベンダーなどが発見した脆弱性について公開しているサイトを確認することで、既知の脆弱性を知ることができます。
2. セキュリティリスク情報データベース: CVEやJVNなど情報セキュリテイ情報を発信するデータベースには、OSSの脆弱性を含む多くセキュリティリスク情報が掲載されています。

参照できる関連情報公開元

* 1. CVE(Common Vulnerabilities and Exposures):https://www.cve.org/
  2. IPA (情報処理推進機構)脆弱性対策HP:http://www.ipa.go.jp/security/vuln/
  3. JVN(Japan Vulnerability Notes) 脆弱性対策情報データベース:http://jvndb.jvn.jp/index.html
  4. JPCERT(Japan Computer Emergency Response Center) CC:https://www.jpcert.or.jp/vh/top.html
  5. OSV(Open Source Vulnerabilities):https://osv.dev/
  6. https://www.cisa.gov/known-exploited-vulnerabilities-catalog

1. セキュリティグループコミュニティへの参加：企業や法人からなるセキュリティグループへの参加を通じて、OSSのセキュリティに関する動向、攻撃手口、最新の情報の入手、関連業種の他会員と解決策を模索することができます。
2. ツールを用いた脆弱性スキャン: セキュリティベンダーが提供するリスク解析ツールや、OSSとして公開されているツールを用いてスキャンすることで、脆弱性を発見することができます。
   1. OSSのセキュリティリスク対応  
      　脆弱性が発見された際は迅速にリリースされた修正パッチを適用することが重要です。

すべての脆弱性へ対応を行うことが難しい場合は、公開されているCVSS(※)の評価結果を用いて脆弱性の影響度をはかり、環境への影響を踏まえたトリアージが有効です。  
  
トリアージは組織の顧客、経営層、開発者などのステークホルダとのコミュニケーションのうえ、対応する脆弱性の優先度付を行うことで、特に緊急時に指摘されたすべてのリスクへの対応が現実的に困難な場合は、許容出来ないリスクを明確化することが必要です。

なお、修正パッチの適用が難しい場合や、パッチ自体がリリースがされない場合は、セキュリティに問題のない代替となるライブラリを検討する必要があります。

※CVSSスコアは、脆弱性の深刻度を0~10.0の範囲で評価した指標です。  
IPA 共通脆弱性評価システムCVSS概説. <https://www.ipa.go.jp/security/vuln/CVSS.html>  
IPA共通脆弱性評価システムCVSS概説によれば、基本評価基準に加え、脆弱性が現在利用可能かどうかを評価するために使用される「現状評価基準」、脆弱性が組織の環境にどの程度影響を及ぼすかを評価するために使用される「環境評価基準」の要素を活用することで組織に応じた効率的な脆弱性対応の優先度を決定することが出来ます。指標の例として、Access Vectorは攻撃がネットワーク経由で行われるか、ローカルからの攻撃が可能かを示す指標です。これにより、攻撃可能範囲や特権が必要かなど、様々な脅威を確認することができます。脆弱性の脅威を正しく理解するためには、CVSS総合スコアにだけ注目せず各要素も考慮することが重要です。

* 1. OSSのセキュリティリスク対策・予防

いくつかの方法で事前にOSSのセキュリティリスクへ対策・予防することが可能です。

* + 1. 環境のバージョン最新化：使用するOSSコンポーネントを定期的に最新バージョンへ置き換え、セキュリティアップデートを適用する。
    2. 定期的な脆弱性のスキャン: 脆弱性スキャンツールを使用して、既知の脆弱性を検出する。

ソフトウェアのセキュリティインシデントに対応する組織、プロセスの設計については、JPCERT/CCから日本語版が公開されている「PSIRT Services Framework」、「PSIRT Maturity Document」が非常に参考になります。このフレームワーク、モデルはOSSを使った製品にも適用することができます。

PSIRT Services Framework：<https://www.jpcert.or.jp/research/psirtSF.html>

* 1. OSSのセキュリティリスクを管理するためのベストプラクティス
     1. 正規のパッケージを利用する

OSSは公式のリポジトリやパッケージマネージャーからインストールすることを推奨します。悪意のあるパッケージ登録者がいる場合、ダウンロードしたパッケージには悪意のあるアルゴリズムがない方されている可能性があります。公式のパッケージマネージャーからインストールすることで、安全性を確保することができます。

* + 1. サポートライフサイクルの理解: 採用するOSSのサポートライフサイクルを理解することでソフトウェアの適切なバージョン選択、関連パッケージの採用、セキュリティアップデートの適用ができ、OSSの採用リスクを軽減することができます。サポートは基本的に最新のメジャーバージョンが対象です。OSSプロジェクトによっては、サポートポリシーやソフトウェアライフサイクルは公式ページに記載されることがあるため、これらの情報の確認が推奨されます。
    2. 脆弱性スキャンツール: セキュリティベンダーが提供するリスク解析ツールや、OSSとして公開されているツールを用いてスキャンを活用して、脆弱性を発見することができます。脆弱性を把握、管理することで、OSSのセキュリティリスクの軽減ができます。
    3. ソフトウェアの脆弱性評価: 脆弱性データベースや脆弱性情報サイトなどから得られるCVSSスコアを活用し、OSSの脆弱性評価を行うことで、ソフトウェアに存在する脆弱性の対応を講じることができます。
    4. 脆弱性管理ガイド、プロセスを策定する：OSSを使用する企業や組織は、使用するソフトウェアに対するセキュリティポリシーやガイドライン、対応プロセスを策定し、運用することでリスクを管理することができます。プロセスについてはPSIRT Services Framework が参考になります。(→4.5 (b)、4.8 (c))
    5. e-ラーニング、トレーニング: OSSを使用する開発者や運用者に対して、セキュリティトレーニングを実施することで、OSSの採用に際しての正しいリスクの理解、脆弱性発見後の早期に対処する能力を養うことができます。
    6. SBOMの活用：SBOM(Software Bill of Materials)とはソフトウェアのコンポーネント、バージョン、OSSライセンス、脆弱性などの情報を記載したドキュメントのことを指します。SBOMを活用することで、上に挙げたコンポーネントの特定や脆弱性への対応などOSSセキュリティリスクを軽減することが期待されています。  
       ※ SBOMはOSSライセンスに関する問題に対しても有効です。SBOMはMachin Readableなファイル形式であり、一般的に使用するためには、SBOMを読み取り・分析するためのツールが必要になります。詳細は3章で説明・解説がされています。
  1. 動向、レポート、ケーススタディ
     1. [OpenSSF 2022 アニュアル レポート](https://www.linuxfoundation.jp/publications/2023/02/openssf-2022-annual-report-jp/)  
        OpenSSF（Open Source Security Foundation）が、2022年に公表した報告書です。このレポートは、OSSセキュリティ、コミュニティの動向、OpenSSFが実施する取り組みなどが掲載されています。
     2. [Synopsysオープンソース·セキュリティ＆リスク分析レポート(OSSRA)](https://www.synopsys.com/ja-jp/software-integrity/resources/analyst-reports/open-source-security-risk-analysis.html)

Synopsys社が公開しているOSSのリスク状況を分析したレポートです。M&Aに際して行われた監査の情報が基となっており、OSSのセキュリティだけではなく、ライセンスコンプライアンスの問題にも焦点が当てられています。

* + 1. [Snyk オープンソースソフトウェアにおけるセキュリティの現状（The State of Open Source Security）](https://prtimes.jp/main/html/rd/p/000000007.000092857.html)

Snyk社が公開しているオープンソースセキュリティの状況を分析するレポートです。オープンソースプロジェクトの脆弱性、セキュリティトレンドなどOSSのセキュリティに関する課題についての情報を提供しています。

* + 1. [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（2022年5月拡充版）](https://www.meti.go.jp/press/2022/05/20220510001/20220510001.html)  
       経済産業省が策定したドキュメントで、オープンソースソフトウェア（OSS）を利用する企業・組織が、セキュリティリスクを軽減しながらOSSを活用する事例を事例集として提供しています。各企業がOSSを選定・評価する際のポイント、OSSの導入・運用・保守における適切な手順、またOSSのライセンスに関する考慮事項などが説明されています。
  1. OSSセキュリティに役立てられるリソース（オンラインコミュニティ、トレーニングなど）
     1. [CIS Software Supply Chain Security Guide](https://www.cisecurity.org/insights/white-papers/cis-software-supply-chain-security-guide)  
        米国の非営利組織であるCIS（Center for Internet Security）が作成したガイドです。  
        ソフトウェアサプライチェーンのセキュリティに関するベストプラクティスを提供することを目的としています。

このガイドは、ソフトウェアサプライチェーンにおけるセキュリティの重要性に焦点を当て、ソフトウェア開発者、サプライヤー、顧客などの関係者に向けた実践的なアドバイス、また内容としてはサプライチェーン攻撃のリスクを軽減するための戦略や、セキュリティのためのフレームワーク、技術的なガイダンスが含まれています。

* + 1. [SLSA（Supply chain Levels for Software Artifacts）](https://slsa.dev/)  
       SLSAは、ソフトウェアサプライチェーンのセキュリティに関するフレームワークの１つです。Googleが開発し、2021年に公開されました。SLSAは、ソフトウェアサプライチェーン内での信頼性を高め、サプライチェーン攻撃から保護するためのベストプラクティスを提供しｒており、OSSの利用者や開発者が、ソフトウェアサプライチェーンの信頼性を高めるための指針としても利用できます。  
       SLSA Whitepaper: SLSAの概要や目的、各レベルの説明、実装例などが記載されています。  
       SLSA Implementation Guide: SLSAの実装についてのガイドです。SLSAを実装する際に参考にできます。  
       SLSA Scorecard: SLSAの各レベルに対する評価ツールです。自社のOSSのSLSAレベルを評価する際に利用できます。
    2. [PSIRT Services Framework 、PSIRT Maturity Document](https://www.jpcert.or.jp/research/psirtSF.html)  
       (4.4) OSSのセキュリティリスク対策・予防でも紹介したPSIRT Services Frameworkは、製品のセキュリティインシデントを処理するためのPSIRTチームのための指針を提供するフレームワークです。OSSを活用する製品においても、このフレームワークは役立てられます。一方でPSIRT Maturity Document は企業や組織のPSIRTの構成、プロセスの成熟度を評価する情報が提供されています。これらはJPCERT/CCから日本語版が公開されています。
  1. ソースソフトウェアのセキュリティに関連する実践的なヒント、Tips、法律、慣例など
     1. 発見したOSSの脆弱性にどう対処すべきか。  
        OSSの脆弱性が発見された場合は、速やかにOSSの開発者やメンテナーに報告することが大切です。報告先について、OSSプロジェクトのWebサイトやリポジトリに報告に関するポリシーや手順が掲載されている場合があります。脆弱性が発見された場合は各OSSプロジェクトの方針を確認しましょう。

報告先が不明な場合は、[Zeroday Initiative](https://www.zerodayinitiative.com/advisories/disclosure_policy/)へ報告することができます。Zeroday InitiativeはTrend Microが運営する脆弱性情報の収集・報告・研究を行うプログラムです。報告することができます。報告された情報を基に、パッチが公開される前に脆弱性を修正するための情報をベンダーに提供します。

* 1. その他