ABB BAILEY JAPAN 9.FEB 2023

可用性を維持した制御システムの継続的改善手法とは?

ABB日本ベーレー　大石貴之
（情報処理安全確保支援士　登録番号 第018981、 ABJ-PSIRT コマンダー）

ABBの現況

グローバル市場における良好なポジション

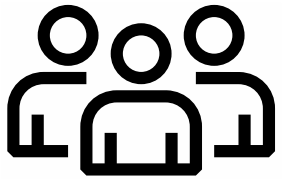
2021 実績

ABBは、より生産的で
持続可能な未来を実現するために、
社会と産業の変革に活力を与える
世界有数のテクノロジー企業です。

エレクトリフィケーション、
モーション、
プロセスオートメーション、
ロボティクス&
ディスクリートオートメーション
のポートフォリオにソフトウェアを接続することで、ABBはテクノロジーの限界を押し広げ、パフォーマンスを新たなレベルに引き上げています。

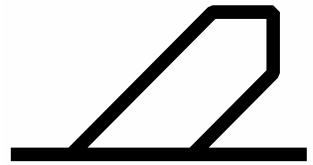
ABB日本ベーレー株式会社

エネルギー産業向けオートメーションのマーケットリーダー



従業員数

約 200 人



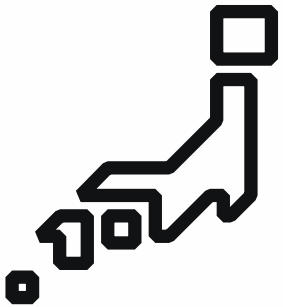
マーケットシェア

50%

- 火力発電所ボイラー制御システム

30%

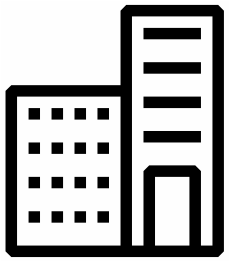
- LNG受入基地制御システム



国内事業拠点

5 拠点

- 本社所在地
  ： **静岡県伊豆の国市**



設立

1971 年

- 日本ベーレー（株）設立
- 2008年　現社名に変更
- ABBの日本国内法人のひとつ

＊2022年7月現在

自己紹介

ABB日本ベーレー　大石貴之

2010 ～ 2018

DCS制御システムの設計・調整・メンテナンス業務を担当

2018 ～

制御システムのセキュリティ設計・検証・ソリューション提案・制御最適化

- 制御システムのハーデニング
- 脆弱性検査
- IDS,EDRの検証
- プロセス制御データサイエンティスト

Agenda

本日お話すること

- 脆弱性をついた攻撃事例
- 継続的なセキュリティ対策はなぜ難しいのか？
- 継続的なセキュリティ対策を実現するには？
- まとめ

脆弱性をついた攻撃事例

事例紹介

国内企業の工場 ランサムウェア

工場のリモート接続機器の脆弱性を突いた攻撃。

ハッカー集団 ロビンフッドによりランサムウェアを設置され、自社だけでなく親会社、更にその取引先まで影響が広がった。

この事例では、サプライチェーンの供給が滞り、国内14工場の28生産ラインが1日ほど停止した。

インドの電力会社 ランサムウェア

一部のITシステムに影響を与えるサイバー攻撃。

ハッカー集団 Hive により
・従業員の個人情報
（住所、電話番号、給与情報）
・エンジアリングデータ（図面）
・財務情報
・顧客情報
といったデータが盗まれ、身代金交渉に失敗したためダークウェブリークサイトに掲載される。

電力の供給には影響なし。

日本企業を主に標的 ウイルス感染メール

日本の業種や企業規模を問わないなりすましメールによる攻撃。

Emotetに感染すると、端末内のメール履歴やアドレスの情報が搾取される。巧妙ななりすましメールを用いて感染を拡大させる。

感染拡大は自組織内にとどまらず取引先にまで影響。

2020年流行したものが再度流行。2021年末以降、多く報告がされている。

Webサーバーなど 脆弱性攻撃

Webサーバーなどに使われているソフトウェアの脆弱性を利用した攻撃。

ソフトウェアは多くの企業・製品で利用されており問題の脆弱性はLog4Shellと呼ばれ、任意のコードをリモートで実行できる。

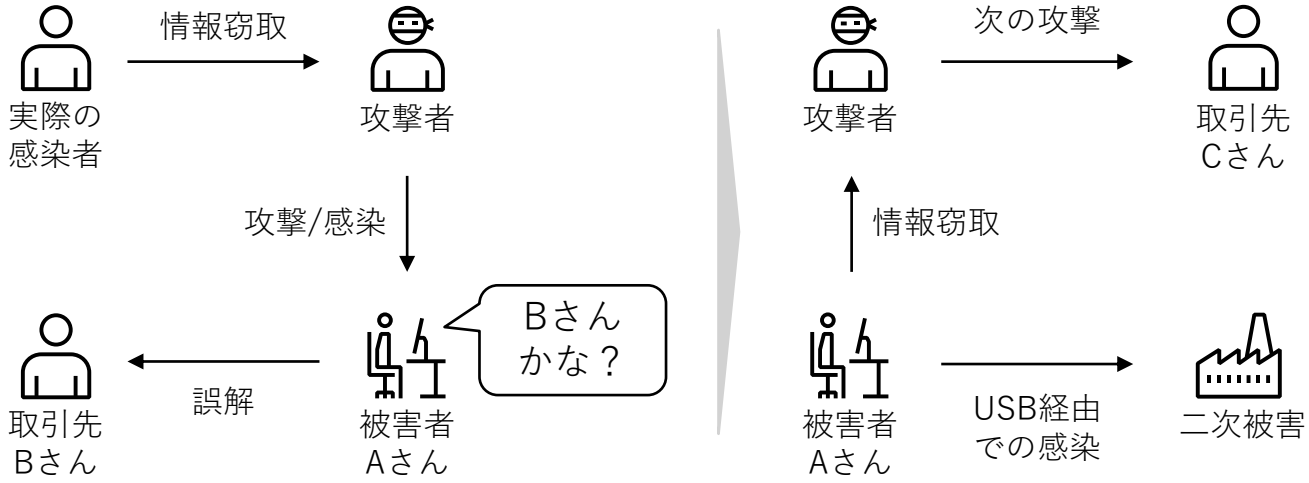
ランサムウェアと組み合わせた攻撃も多い。2021年末以降、多くの報告がされている。

誰もが標的であり、攻撃者に加担してしまう攻撃

情報の窃取と、さらなるウイルス感染のために用いられているウイルス Emotet

特徴

- メールに添付されたWordやExcelファイルなどを活用し、対象を感染させる
- あたかも相手からの返信のように装い、かつ正規のメールに紛れるように送付されるため不正なメールだと気づきにくい
- Emotet の感染により情報が窃取され、新たな攻撃メールの材料とされてしまう
- Emotet から他のウイルスに感染。ランサムウェアに感染すると、ファイルが暗号化されて起動できない状態になってしまう
- メール本文中のURLリンク、パスワード付きzipファイルなどウイルス対策ソフトの検知をすり抜けさせる手口が巧妙化



例えば、制御システムベンダーの名前で「このファイルを制御システムの保守ツールで実行してください」という文面のメールが届き、作業員がUSBに添付ファイルに保存する。USBに対してウイルスチェックしても検知されず、保守ツールでファイルを実行、感染させてしまう。

対策

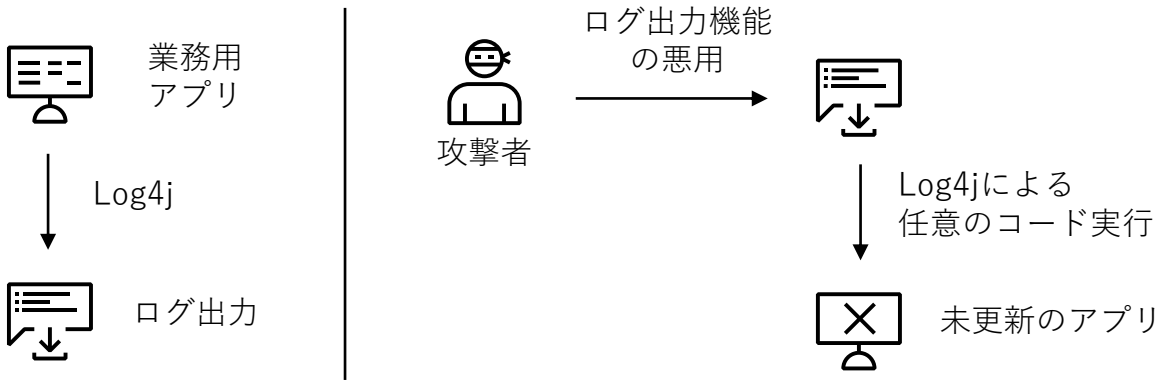
- **Office製品のマクロ実行 無効化**
- **不用意に添付ファイルやURLを開かない**

Webサーバーの脆弱性を利用した、リモート実行攻撃

ログ出力ライブラリ Apache Log4j の脆弱性 Log4Shell

特徴

- Log4j は、オープンソースプロダクトであるログ出力ライブラリ
- Log4j は、 Web サーバー以外にも業務用アプリケーションなど多くの製品・システムで用いられている
- 発見された脆弱性 Log4Shell は、攻撃者が任意のコードをリモートで実行できてしまうもの
- 通常処理となってしまうため、ホワイトリストによる実行防止やパターンマッチのウィルス対策ソフトによる検知・駆除が難しい
- ユーザ側はこの機能が使用されているかを判断することも難しい
- ランサムウェアと組み合わせた攻撃も多い



対策

- **WAF、IPS/IDSによる検知、遮断**
- **ソフトウェアをベンダーの提供する最新版へアップデートする**

「どうするべきか？」

攻撃者やマルウェアを侵入させないことが最大の対策です。
ただし、自分自身が拾ってしまうこともあれば、プログラムの動作として取りに行ってしまうこともあります。
意図せずマルウェアが持ち込まれてしまったとき、それが被害に発展するかは、設備が如何に対策されているかに他なりません。

・Webアプリケーションファイヤーウォール
・IDS/IPS
・マルウェア対策ソフト
・セキュリティアップデート

このような対策は攻撃手法に対してアップデートが非常に有効になります。
今問題なく動いている設備を触るのは、とても勇気がいるかもしれません。
メーカやエンジニアの立場でも直面しています。
ですが、継続的な対策は絶対に必要なことだと思っています。

継続的な
セキュリティ対策は
なぜ難しいのか？

サイバーセキュリティ対策の継続的改善
改善要素と現在最も効果的な対策

継続的改善要素

- OSのセキュリティアップデートの更新
- ウイルス対策ソフトの定義ファイルの更新
- DCSやHMI周りソフトウェアの定期的なアップデート
- 周辺機器(ルータ、ファイアウォール、エッジデバイス、プロトコルゲートウェイ等)のファームウェア更新
- コンピュータ(工業用、非工業用問わず)のBIOS/UEFIの更新
- セキュリティインシデント情報共有/積極的な情報収集

現在最も効果的な対策

脆弱性管理を効率的に。厳格に。

- セキュリティパッチのサーバーを設置
- ウイルス定義ファイル/OS・ミドルウェアのセキュリティアップデート
- サードパーティーSWのセキュリティアップデート
- ファームウェア/ハードウェアの更新

セキュリティインシデント情報共有/積極的な情報収集

- ベンダーから情報を仕入れ、定期的な更新の有無を確認していく
- 定期的な脆弱性検査を実施、アラート発信
- ベンダー推奨設定との比較

サポート期限に合わせた設備更新

- OSの更新
- ミドルウェアの更新
- HMIシステムのバージョンアップ
- ハードウェアの更新

制御システムに適用される代表的なセキュリティ対策の強み
被害を防ぐには、侵入させない、実行させない、成功させない。

OSアップデート、セキュリティアップデート

- 脆弱性を悪用する攻撃は広く普及したOS、Office、OSSをターゲットにしたものが多い
- マルウェアの侵入を許し、攻撃実行された場合でも、ターゲットとなるソフトウェアや環境がアップデートされていることで被害を防ぐことが可能
- OSの更新を含む定期的なセキュリティアップデートは被害を未然に防ぐ、防御の点では最も重要なセキュリティ対策

ウィルス定義ファイルの更新、検査

- パターンマッチ型のウィルス対策ソフトは定義ファイルを常に最新に保つことで、効果が維持できる
- マルウェアにシステム内に侵入された場合でも、実行前の検査で検知、駆除できれば被害を未然に防げる。

周辺機器（ルータ、ファイアウォール、エッジデバイス、プロトコルゲートウェイなど）のファームウェア更新

- 攻撃者が狙うのはコンピュータだけに留まらない
  ルータ、ファイアウォール等の設備は組み込みデバイス用汎用OSで構成される場合も多い
- 脆弱性を利用した攻撃では細かく構成されたファイアウォール定義も無効にされる場合がある

なぜ継続的改善が難しいのか？
セキュリティのベストプラクティスは必ずしもノーリスクではない。

OSのセキュリティアップデート

・互換性を維持のために古いライブラリやコンポーネントを捨てきれない

・ソフトウェアのVerが古く、 OSのセキュリティアップデートをサポートしない（同時にソフトウェアのアップデートが必要）

ウイルス定義ファイル

・必要なプログラムがウィルスとして検出、削除されてしまう

・スキャン動作によりコンピュータ負荷が上昇し、通常の処理が遅くなる

サードパーティーソフトウェアのセキュリティアップデート

・現代的なソフトウェア開発においてOSSが広く利用されているが、マイナーバージョンアップに対しサポートしきれない

・サードパーティSWの更新により、メインプログラムに支障が出る可能性は捨てきれない

ファームウェア/ハードウェアの更新

・あるコンピュータで使用されている内部のコンポーネントが通知無しで変更された結果、機能が動作しなくなった

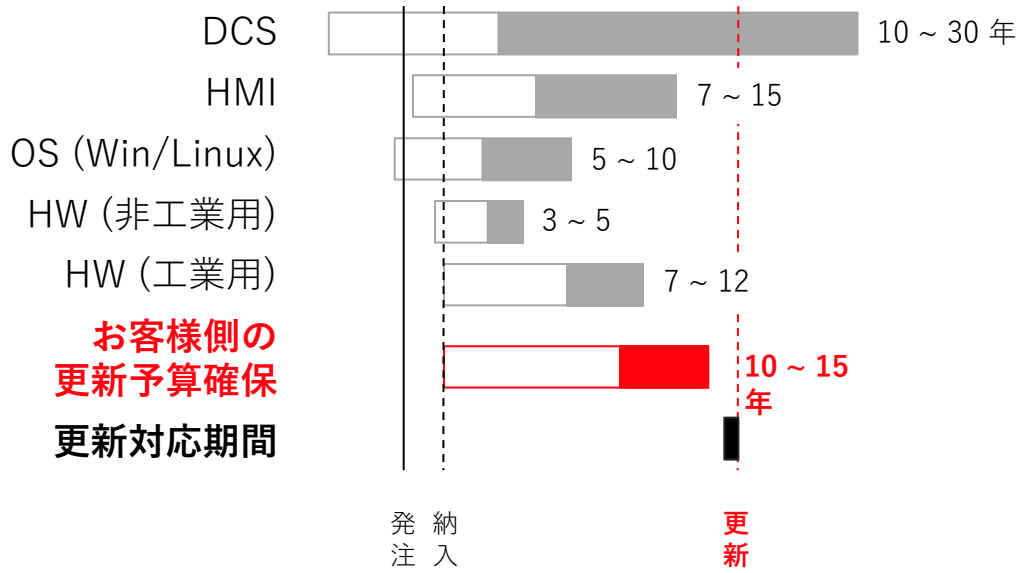
・ドライバが変わることでメモリリークが起きるようになった

・ソースコードは変わってないのに動作が変わってしまった

検証せずにアップデートはできない

なぜ継続的改善が難しいのか？
設備更新のタイミングはハードウェアの寿命。

一般的なDCS、HMIシステム、ハードウェアのライフサイクル



更新計画の難しさ

- 小分けにし、高頻度で更新を実施すれば 1回あたりのコストは少ないが、ライフサイクル全体では…
- ハードウェアとソフトウェアのライフサイクルが異なる
- OSのサポート終了はハードウェアの寿命ではない
- 予算や作業期間の確保が難しく、後回しになりがち
- 結果的に15年に1度、1週間程度で更新をかけるという強行スケジュールになってしまう

なぜ継続的改善が難しいのか？

制御システムベンダー：各セキュリティ対策への課題

システムのセキュリティパッチ、ウイルス定義ファイル
更新への課題

- ソフトウェア的な組み合わせ試験のパターン爆発
- 網羅しきれないパターンへの不安感
  あるバグ修正の影響範囲の把握の難しさ
- 万が一があった場合の対応の難しさ（契約面、心理面）
- ハードウェアとの相性問題

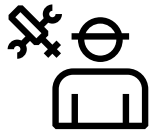
⇒ **確実な検証を実施するために、
製品の保証期限やソフトウェアのサポート終了期限
を設定せざるを得ない**

制御システムのセキュリティ対策への課題

- 脆弱性が発見された場合の迅速な適用の難しさ
- ファームウェア、ソフトウェアのパッチを公開しても
  ITシステムの様に即座に適用を行える環境が非常に少ない
  - 「隔離されてるから大丈夫」
  - 「運用止められない」
  - 「コストがベンダー持ちなら」
  - 「日常的に行っている作業ではない」

⇒ **情報提供のみ
実施判断はユーザー側にゆだねざるを得ない**

なぜ継続的改善が難しいのか？



ベンダーの保守担当者：安定運用・保守への課題

ベンダーの保守担当者の抱える、セキュリティ改善への不安

- 何もしなければそのまま動き続けてくれる。
- お客様が求めているのは安定稼働。
- どちらかといえば現場側に寄りそう意識が強く、問題が起きる可能性があることは極力したくない。

毎月、毎日のアップデートのために毎回現地に赴くことはできない

なぜ継続的改善が難しいのか？

ユーザー：保守と改善の板挟み

セキュリティ改善への苦悩

- ネットワーク的に隔離されてるから、運用を気を付ければ大丈夫という意識がある
- 持ち込ませなければ被害が発生しない
- セキュリティ対策は必須と言われるが、保証の問題がありベンダー任せになってしまう
- **そのうえ、ベンダーもセキュリティアップデートは強く推奨していない**
- 今の運用を維持するための保守予算しかない。追加でコストがかかるのは厳しく、時間もない
- この手の業務がわかる人材の確保が難しい

毎月、毎日のアップデートに毎回メーカ技術者を呼ぶことはできない

自分で実施するとして、作業時の影響範囲がわからない（制御装置の再起動は必要か？運転に制約が必要か？）

など、主に運用方法を変えたくない/変えられないという状況と「セキュリティ対策」というプレッシャーの板挟みになっている

なぜ継続的改善が難しいのか？

制御システムベンダー

毎日の確認作業で手一杯。
パッチ配信はできるけど
現場への適用までは不可能

ベンダーの保守担当者

セキュリティ更新の必要性
はわかるけど、頻度が多く
強く言えない…

ユーザー

更新はたいへんそうだし
強いリクエストがない。
今のままで十分なのでは？

セキュリティ対策を先延ばしにされた
制御システム

攻撃者

ウイルス定義ファイル10年前だ。
脆弱性だらけで何でもできちゃう。
ラッキー！楽勝！

継続的な
セキュリティ対策を
実現するには？

継続的改善を実現するには
正しくリスクを把握、安全なアップデートを実施

OSのセキュリティアップデート

・制御装置ベンダーが検証を行ったサポートを利用する
　サポートされていないアップデートは絶対に行わない

・パッチの適用が遅れた場合でも、ベンダーから提供される検証情報を元に、かならず安全が担保されたパッチを適用する

・必要な場合は、HMIソフトウェアのアップデートも行う

ウイルス定義ファイル

・制御装置ベンダーが検証を行ったサポートを利用する
　サポートされていないアップデートは絶対に行わない

・スキャン設定がベンダー推奨設定になっているか確認する

・オンアクセススキャンを行う
　例）検出時の動作はレポートのみで自動駆除しない など

サードパーティーソフトウェアのセキュリティアップデート

・自社のシステム特有のものは、更新できない可能性もある

・制御装置ベンダーに確認し、更新ができない場合は回避策を検討する

・対策は一つだけではない

ファームウェア/ハードウェアの更新

・セキュリティアップデートの情報と合わせて、ハードウェアのサポート期間を確認する

・ニュースレターなどを受け取り、内容を確認する

・ハードウェアの更新時期の検討と合わせてメーカーに相談する

正しく検証されたアップデートは安全である

継続的改善に必要な要素とは
ベンダーとユーザーの協業

対応者

ベンダー　システム更新（推奨頻度10～15年）

ベンダー　コンピュータ更新又はOSアップデート（推奨頻度7～10年）

ベンダー　・・・OS　メジャーアップデート（推奨頻度0.5年～）

ベンダー：検証情報
作業手順
作業制約情報　の提供

・・・セキュリティパッチ更新（推奨頻度1カ月～）

ユーザー：作業の実施

・・・ウィルス定義ファイル更新（推奨頻度1日～）

ベンダーはパッチの検証、リスク評価、作業手順及び作業時の制約に対して責任を持つ
ユーザは日常保守作業として細かなセキュリティ更新を行う

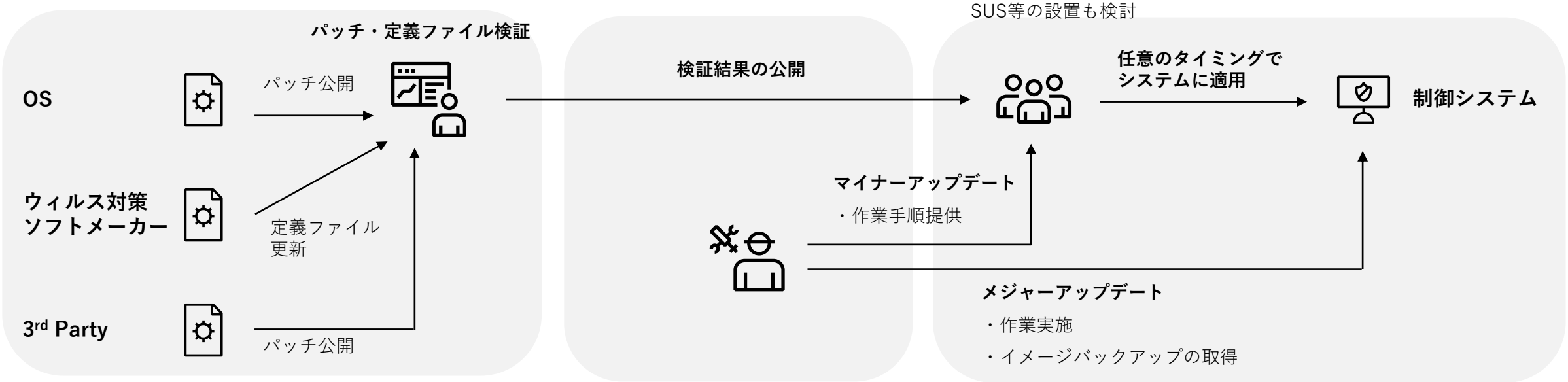
継続的改善を実現するには

制御システムベンダー
検証に注力

ベンダーの保守担当者
安全な適用作業に注力

ユーザー
マイナーアップデートはユーザー自身で適用

適用数が多いサイトでは
SUS等の設置も検討



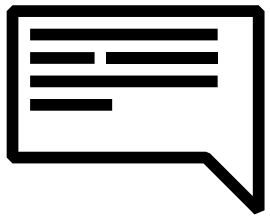
制御システムベンダー、ベンダー保守担当者、ユーザーの協業

まとめ

まとめ
可用性を維持した制御システムの継続的改善手法とは?

検証	• 制御システムベンダーが、製品のライフサイクルの範囲内でパッチの検証を実施 • 安全なセキュリティアップデートをユーザーに提供する
判断	• 検証の結果によって適用できないアップデートが見つかる場合もある • メインプログラムやOSを合わせて更新するか、回避策を実施するかをリスクの保持も含めて判断することが必要
協力作業	• 継続的改善は少量頻回の保守作業 • 制御システムの保守に関わる全員で、役割分担を相談する
契約	• 必要に応じて、メーカ保守内容も見直すことも重要 • 契約内容の確認を行う

セキュリティ対策の継続的改善のすすめ



セキュリティは日常のメンテナンス。
手順通りに。欠かさずに。

協力して維持していきましょう。


ABB