# スマート工場で見過ごされている セキュリティリスク

## ～ミラノ工科大学との共同実証実験に基づく3つの侵入経路と攻撃シナリオ～

トレンドマイクロ株式会社　グローバルIoTマーケティング室
セキュリティエバンジェリスト
石原 陽平（いしはら ようへい）

# 本日のアジェンダ

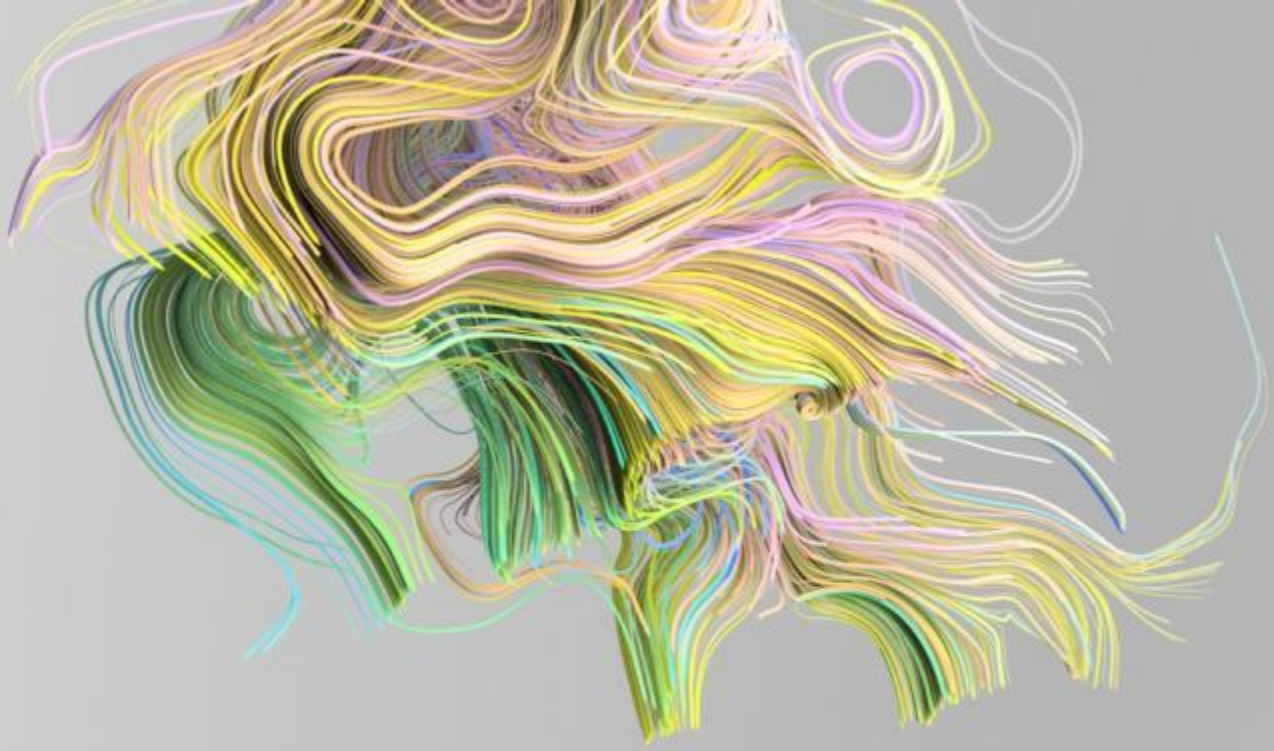
1.工場の『スマート化』とは

2.研究の目的

3.研究手法の特徴

4.結果

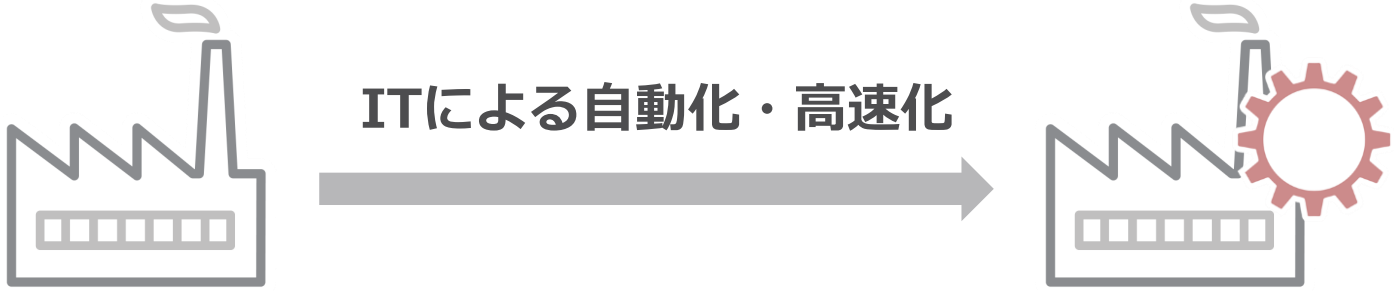
5.推奨されるセキュリティ戦略



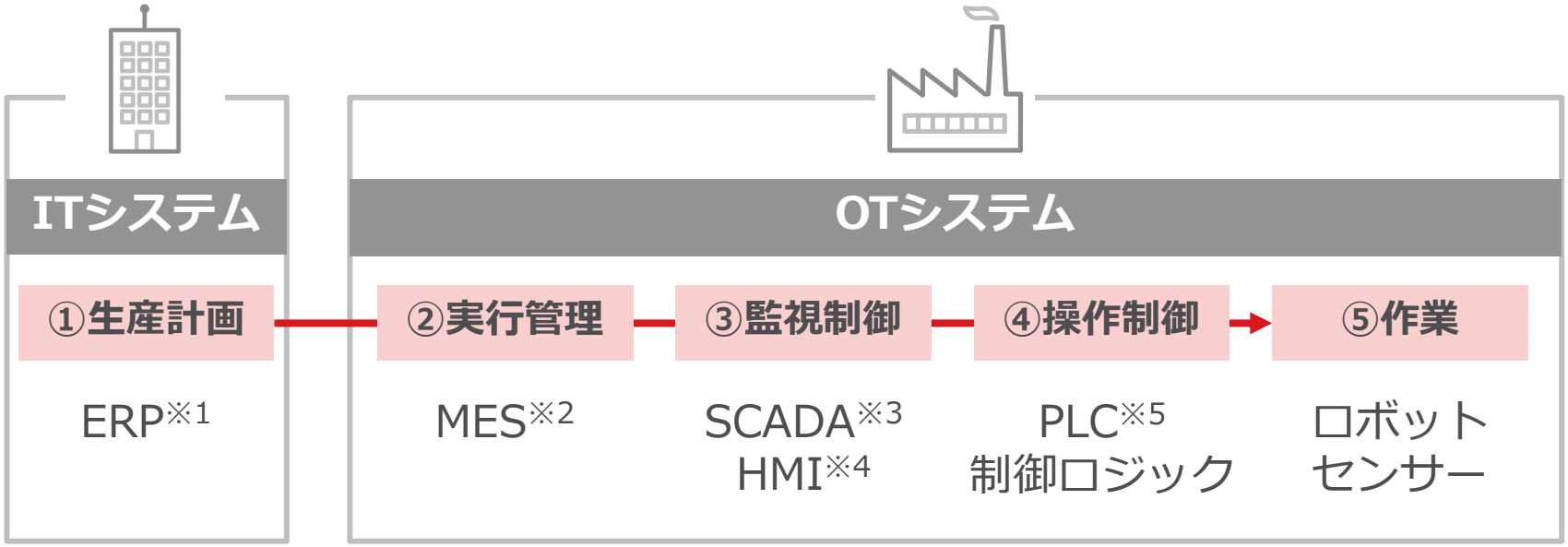
# 工場の『スマート化』とは

# 『工場のスマート化』の定義

**生産性向上**と**不良率低下**の実現を目的とした**IT技術**の活用アプローチ。
具体的には、ソフトウェア制御、ネットワーク接続、そしてデータ分析を駆使することにより、生産活動およびその管理を高度に**自動化**および**高速化**する取り組みのこと。




# データ連携で製造プロセスをより効率的/柔軟に




※1 Enterprise Resources Planning:
会社全体の資源を管理するための統合型システム。

※2 Manufacturing Execution System(製造実行システム):
製造プロセスの状態の把握や管理、作業者への指示や支援などを行う情報システム。

※3 Supervisory Control And Data Acquisition :
ICS（Industrial Control System：産業制御システム）のシステム監視とプロセス制御をコンピュータで行うシステム。

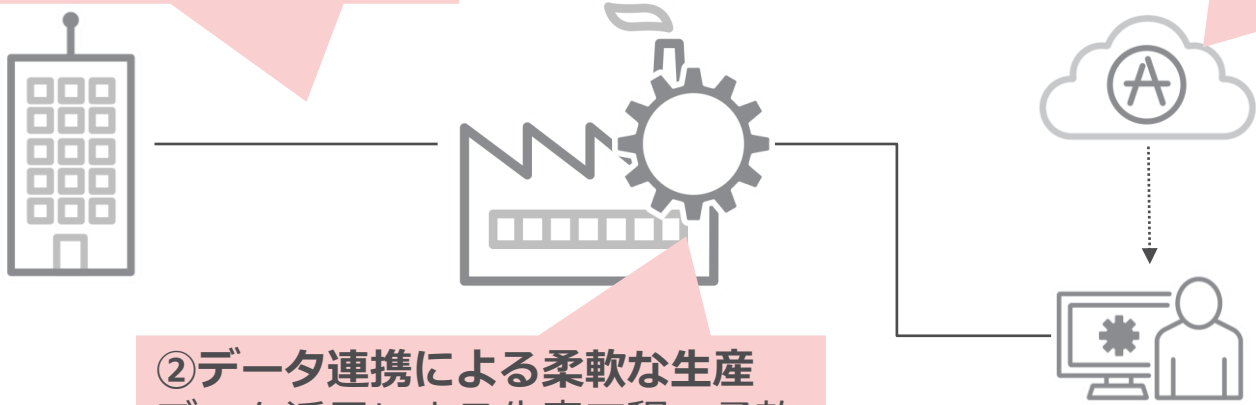
※4 Human Machine Interface :
システム管理者やオペレーターがシステム全体の状況を確認したり、制御したりするためのインターフェース。

※5 Programmable Logic Controller :
ICSで使用される機械を制御するための装置。

# 【例】工場スマート化の取り組み

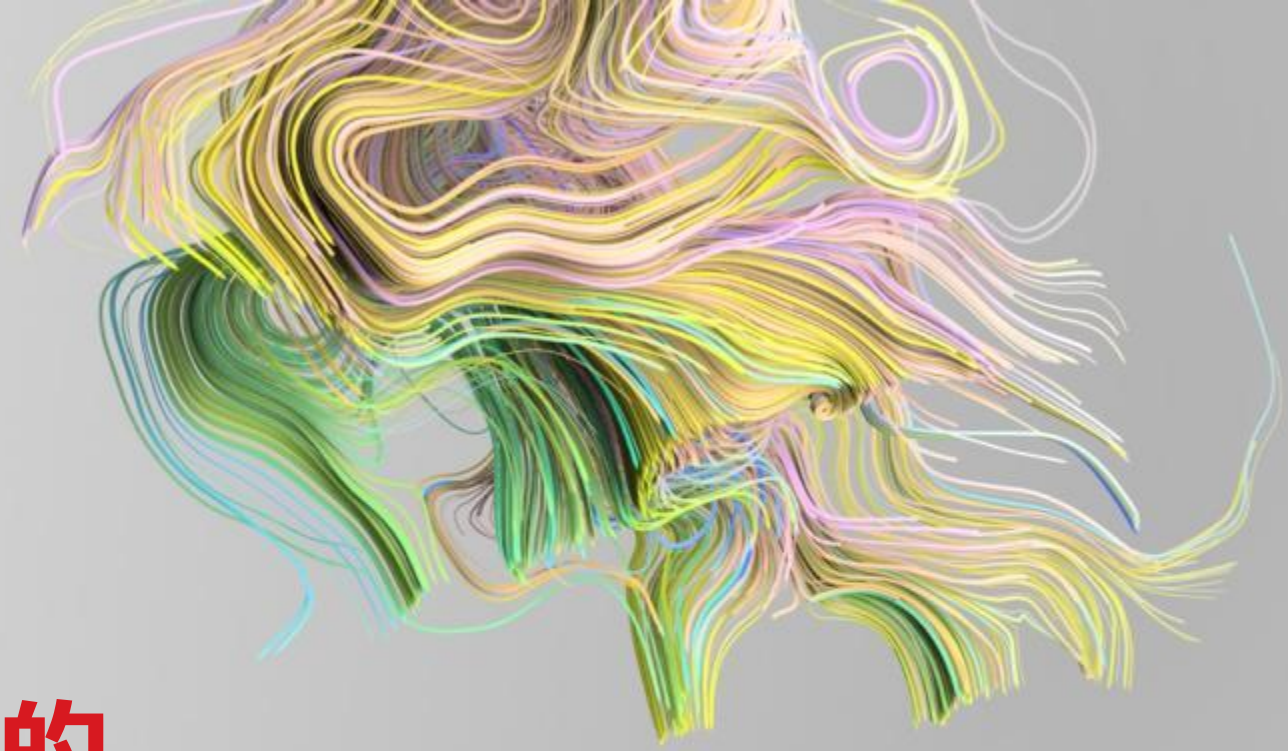
**①IT/OTデータ連携による効率化**
ビジネスデータをリアルタイムに生産活動に反映し生産効率UP

**③クラウド活用による開発高速化**
アプリケーションストアやオープンソース活用による開発期間の短縮



**②データ連携による柔軟な生産**
データ活用による生産工程の柔軟な変更や、故障の未然防止

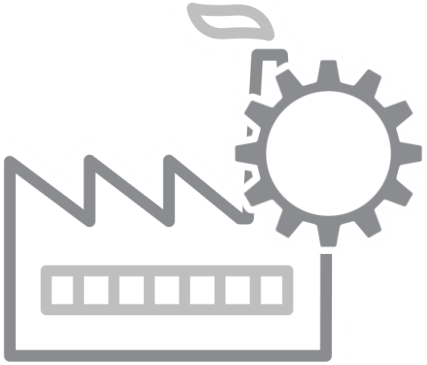
# その一方で…工場のスマート化に伴う
# サイバーセキュリティリスクの増加が懸念される



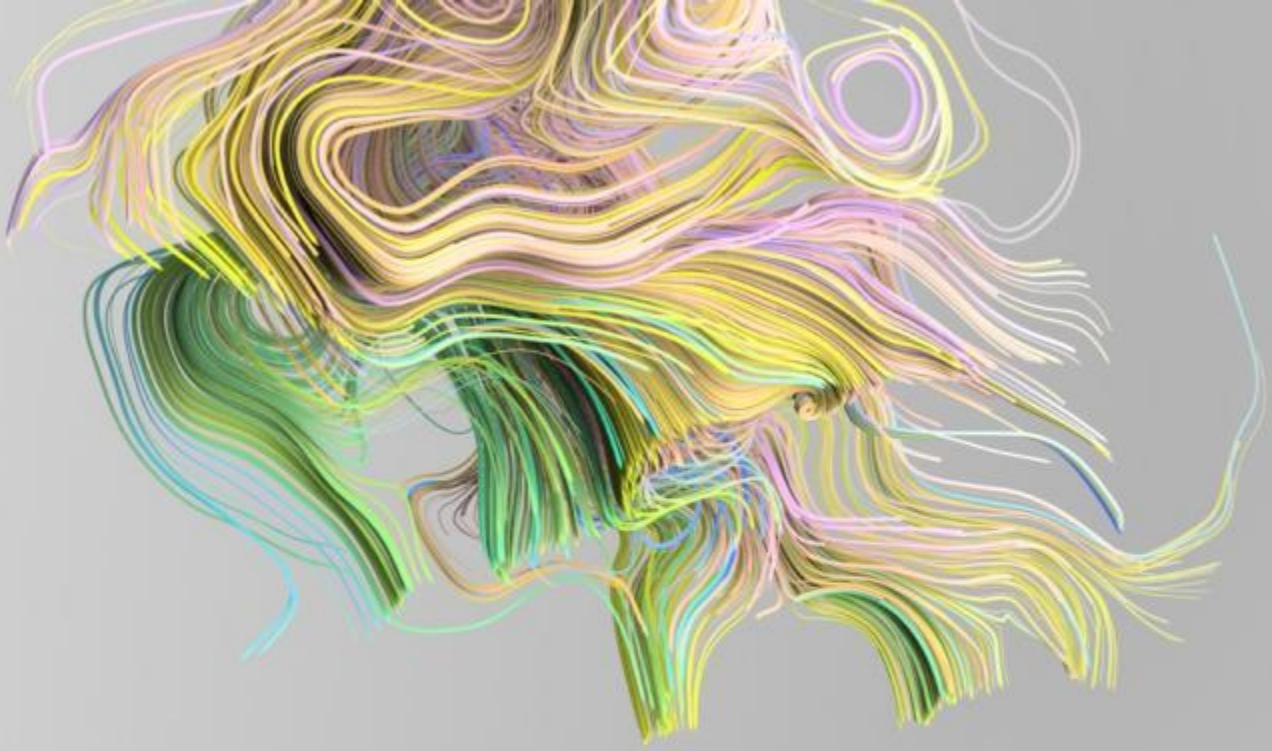
# 研究の目的

# 工場のスマート化を進める際に
# 考慮しておくべきセキュリティリスクを
# 明示する

# スマート工場環境において
# 『見過ごされているセキュリティリスク』は何か？

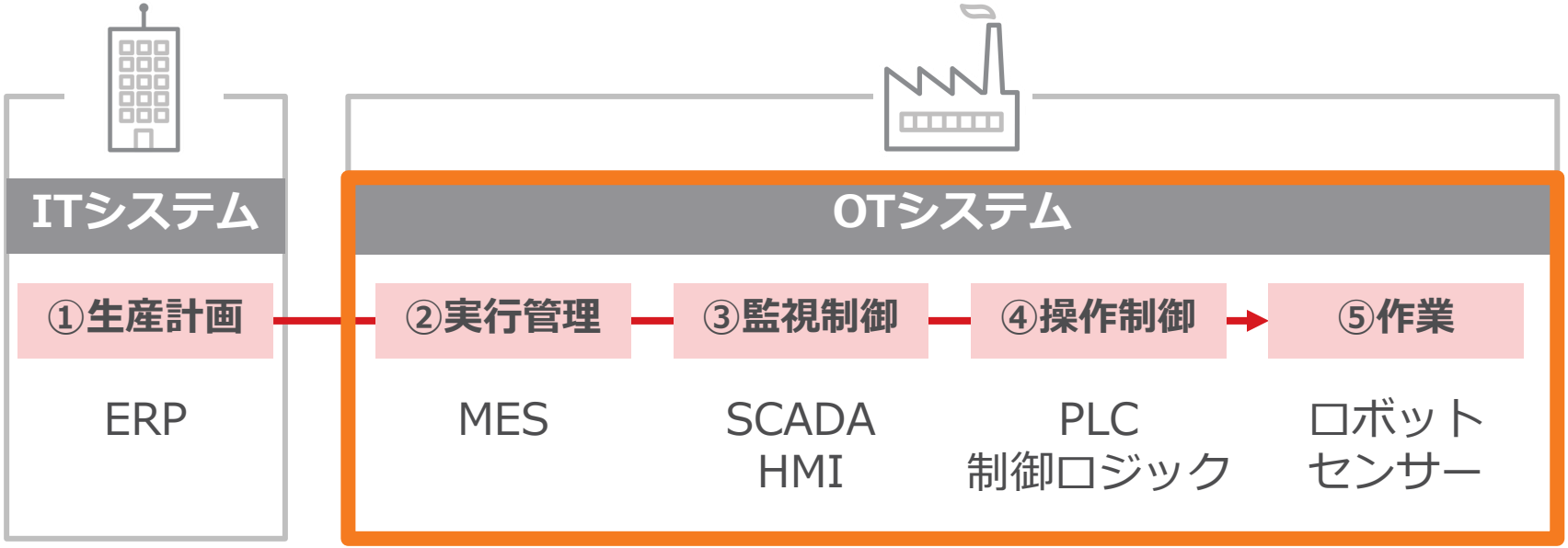


- 新たな**侵入経路**はあるか？
- どのような**攻撃手法**があるか？
- どのような**被害**が出るのか？
- どのような**対策**が有効なのか？



# 研究手法の特徴

# OTシステム内のスマート製造システムを構築

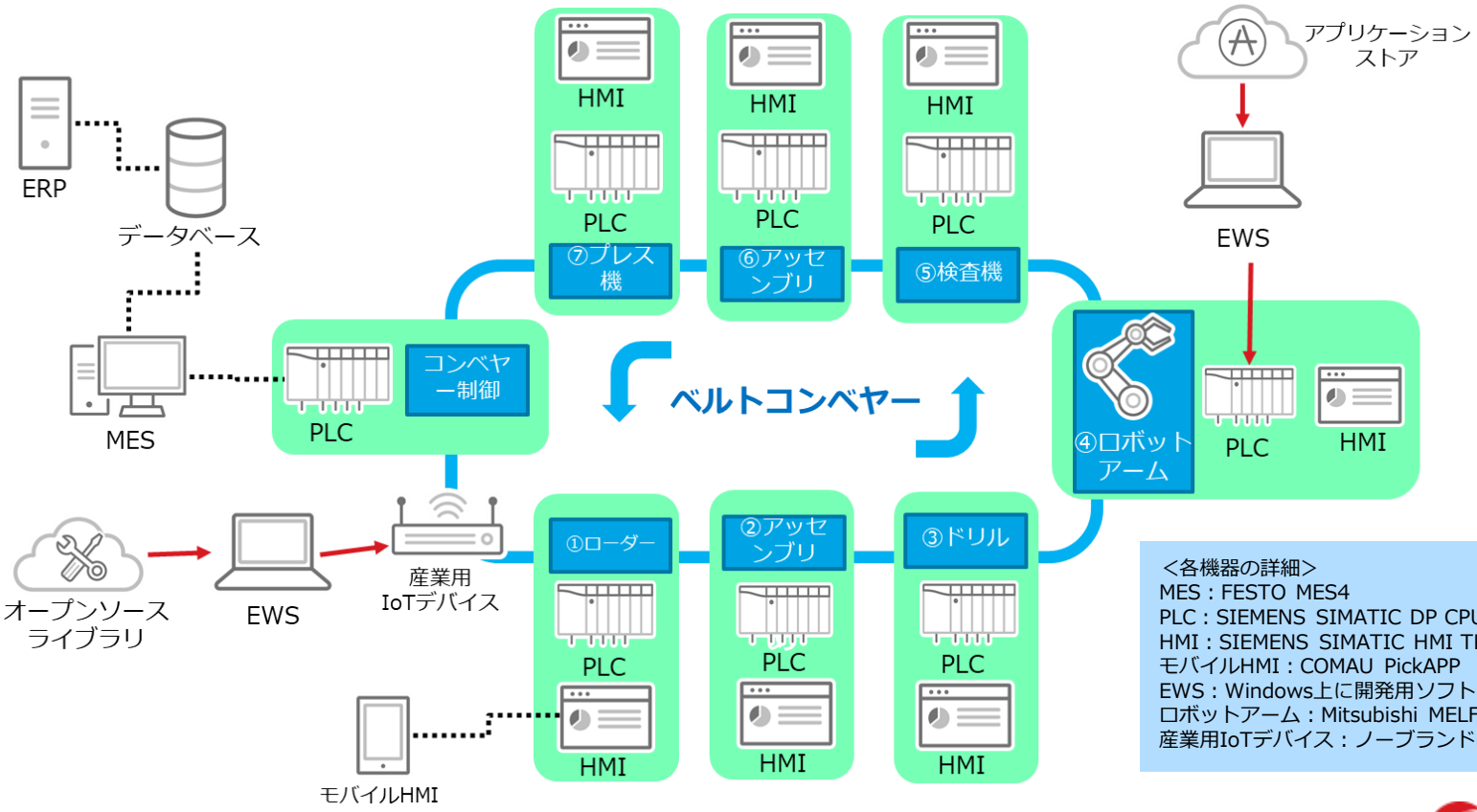



**実環境を構築**

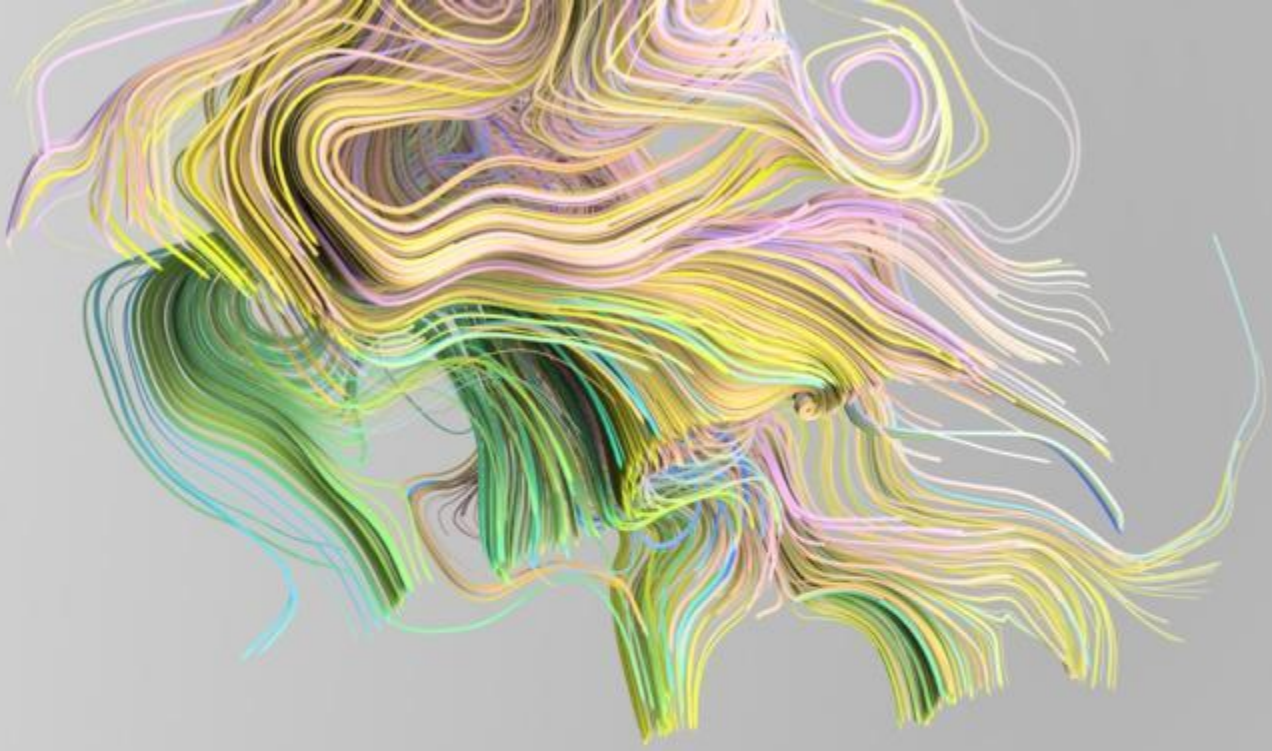
# ミラノ工科大学と共同で実証実験を実施

■写真：本リサーチで使用したスマート製造システム

# 検証環境のイメージ図




<各機器の詳細>
MES：FESTO MES4
PLC：SIEMENS SIMATIC DP CPU 1510SP-1 PN6
HMI：SIEMENS SIMATIC HMI TP700 Comfort7
モバイルHMI：COMAU PickAPP
EWS：Windows上に開発用ソフトウェアを稼働
ロボットアーム：Mitsubishi MELFA V-2AJ8
産業用IoTデバイス：ノーブランド品



# 実証実験の結果

# 注視すべき複数の新たな侵入経路が判明

**スマート化に伴う侵入経路**

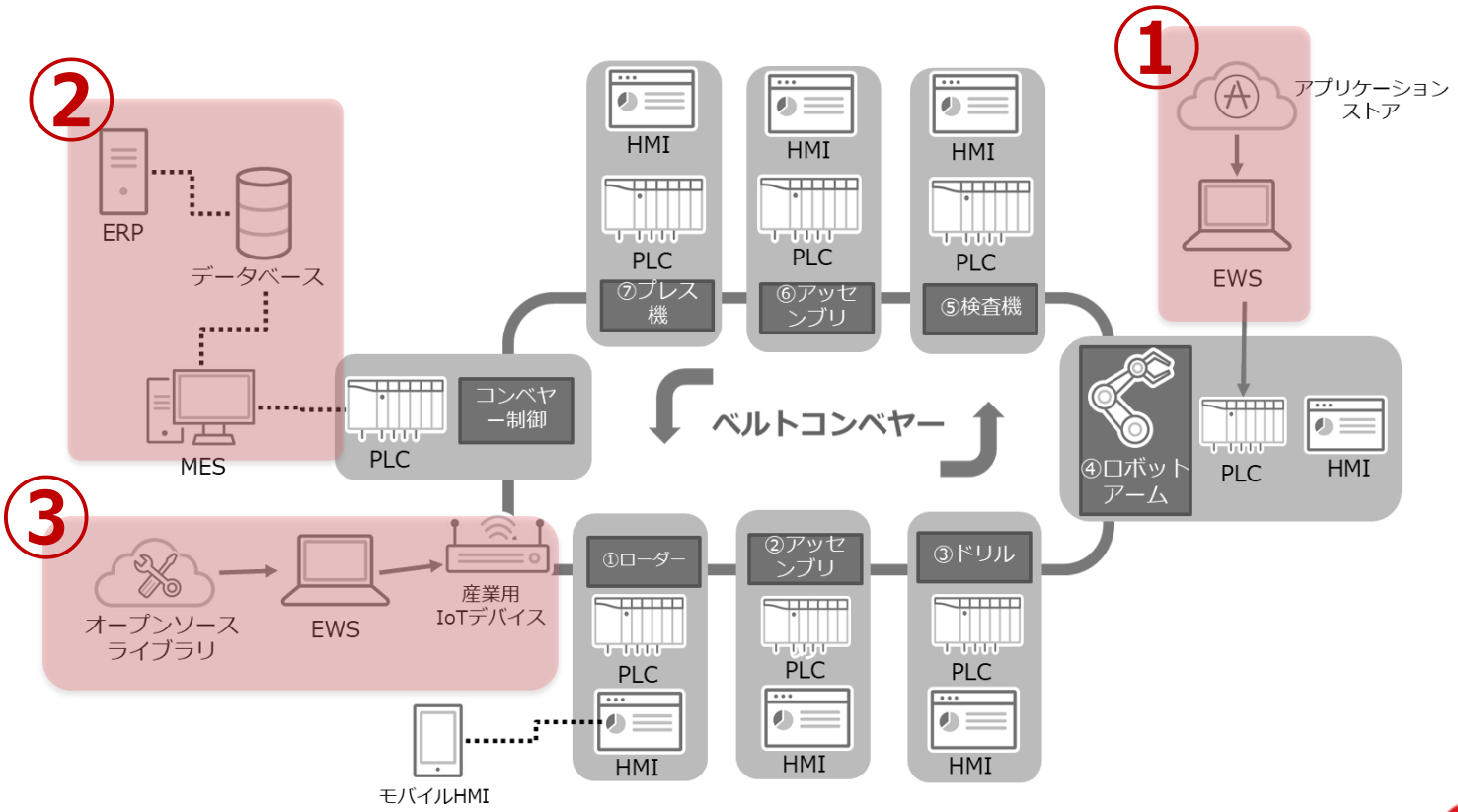
×

**攻撃の実現可能性**

工場のスマート化
に伴って発生する経路かどうか

工場の生産活動に深刻な影響を及ぼす脆弱性
および攻撃手法が実証された侵入経路かどうか

# スマート工場で危惧される三大侵入経路



# 本研究で実証された攻撃シナリオ

1. 産業用のアプリケーションストアの**脆弱性**を悪用したEWSの侵害

2. MESデータベース改ざんによる**製造不良**

3. 悪意あるオープンソース**ライブラリ**による産業用IoT（IIoT）デバイスの侵害

# 攻撃シナリオ①：
# 産業用のアプリケーションストアの脆弱性を悪用したEWSの侵害

## <侵入経路>

アプリケーションストア→EWS→工場全体

## <攻撃手法>

アプリケーションストアに
マルウェアをアップロード

↓

ユーザがEWSに
該当プログラムをダウンロード

↓

マルウェアが
EWS上で実行される

↓

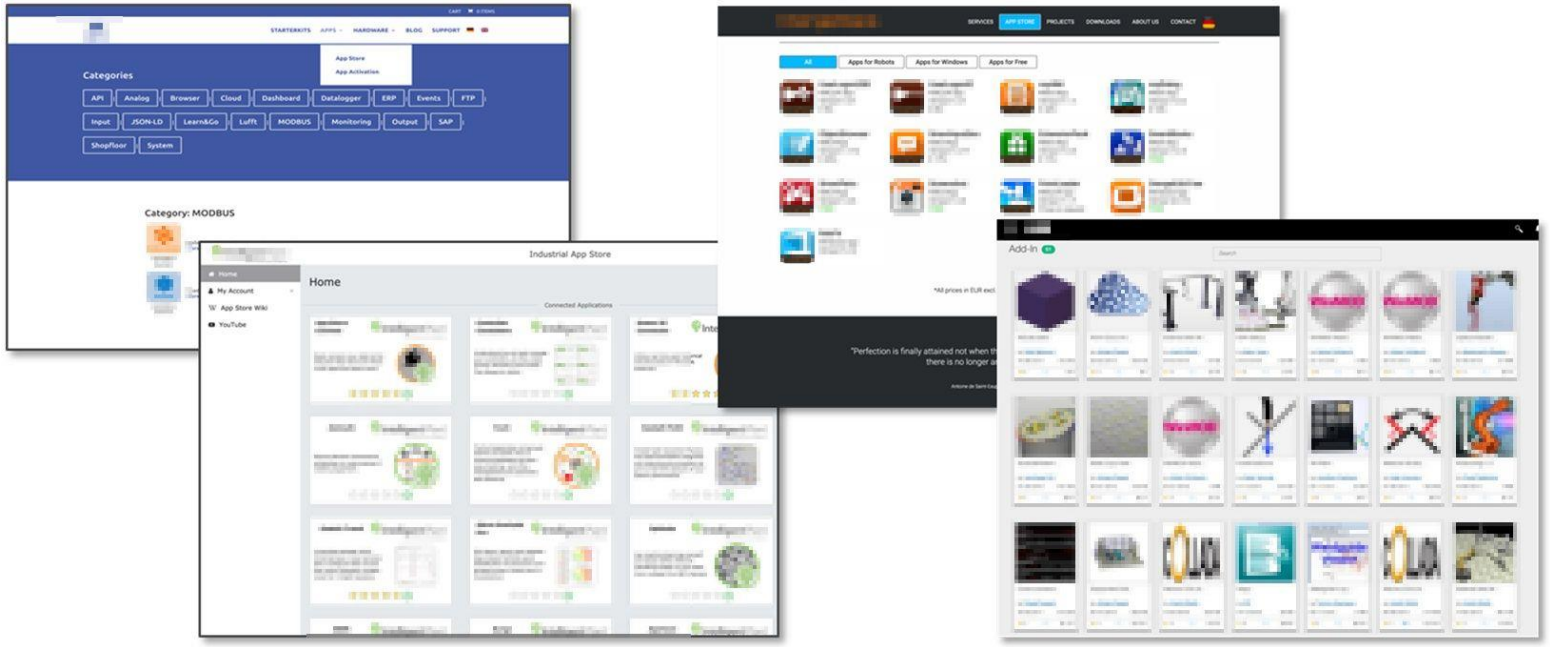
攻撃者による工場への侵入

# EWSの特性：データ交換が頻繁に行われる

1. 単一のEWSが複数の開発者で共有・利用されるケースがある。
2. EWSに直接または間接的に関わる開発者は、社内従業員だけでなく、外部の事業者であるSIerやコンサルタントが利用するケースもある。
3. 外部で作成したプログラムを配布するために利用されることもある。
4. 工場内フロアネットワークに接続されていることが多い。

複数の「人」「データ」が関わり、
工場外部にも内部にも接続される機会があるデバイス

# 産業用のアプリケーションストアは
# スマート化の一つのキーワード

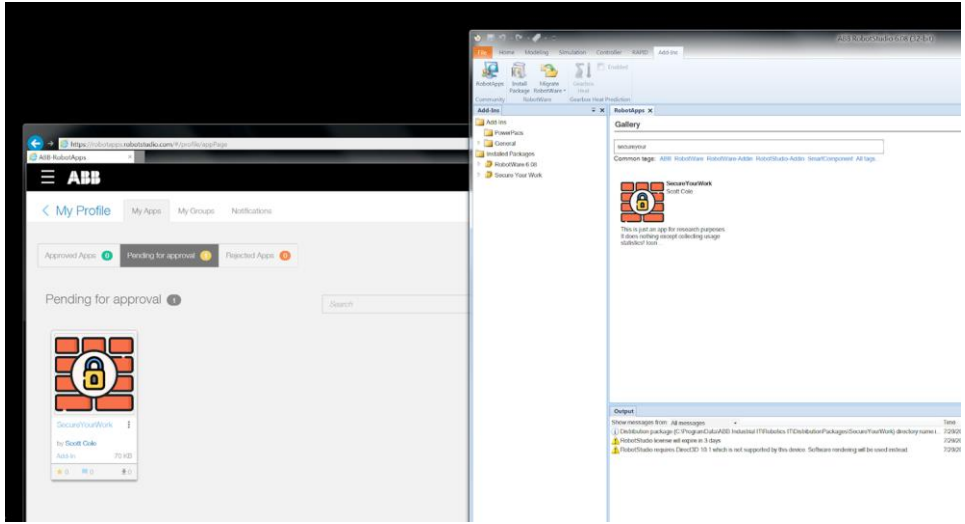



■画面：産業用機器メーカ各社のアプリケーションストアの画面

# 発見された脆弱性①：
# 誰でも自作アプリをアップロード可能かつ承認前にダウンロード可能

- ABB社のアプリケーションストアに脆弱性を発見（ZDI※経由で通知し、本脆弱性は修正済み）。

- 当社が作成したアプリケーションをアップロードしたところ、「Pending for approval（承認待ち）」というステータス表示にも関わらず、ユーザが当該アプリをダウンロード可能な状態であることを確認。

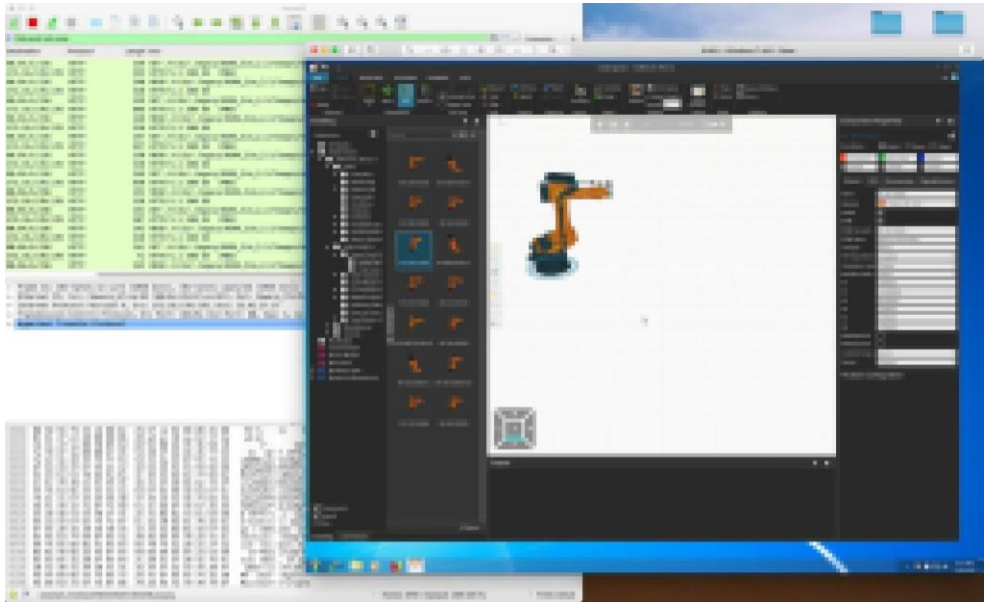



■画面：検証用アプリをABBのアプリケーションストアからEWSにダウンロードした状態

※ トレンドマイクロが運営する脆弱性発見コミュニティ「Zero Day Initiative」。

# 発見された脆弱性②：
# デジタルツインのデータ改ざんが可能かつ通信の暗号化未処理

- KUKA社は「eCatalog」というデジタルツイン※作成機能を提供している。
- アプリケーションレベルでの完全性チェック機能がないため、デジタルツインの改ざんが可能。結果的に、生産ラインの予期せぬ挙動につながる可能性がある。
- 通信もHTTPしかサポートしておらず、セキュリティリスクが高い。



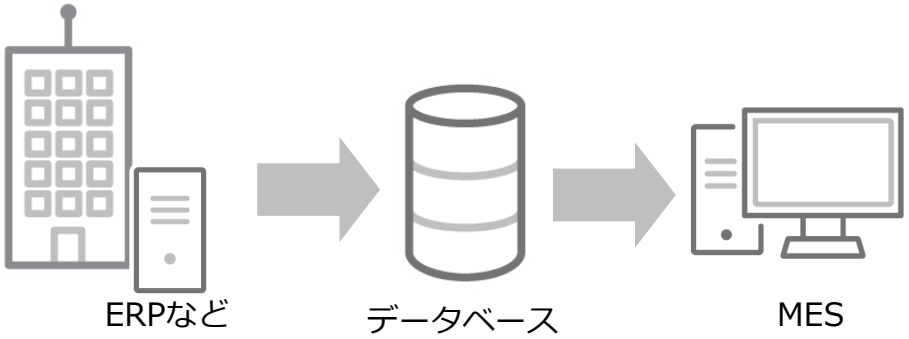
■画面：KUKAの「eCatalog」画面

※ サイバー空間上に物理情報を再現しシュミレーションを行うためのシステム。

# 攻撃シナリオ②：
# MESデータ改ざんによる製造不良

**<侵入経路>**

ITシステム→MES




**<攻撃手法>**

ITシステムから
MESデータベースに侵入

MESのデータベースの
数値を改ざん

PLCのロジックが
変更される

意図しない生産物が
製造される

# 【デモ動画】

# MESデータベース値の改ざんによる製造不良

# 攻撃シナリオ③：
# オープンソースライブラリの改ざんによる産業用IoTデバイスの侵害

## <侵入経路>

オープンソースライブラリ
→EWS→産業用IoTデバイス

## <攻撃手法>

オープンソースライブラリに
マルウェアをアップロード

↓

開発者がEWSに
プログラムをダウンロード

↓

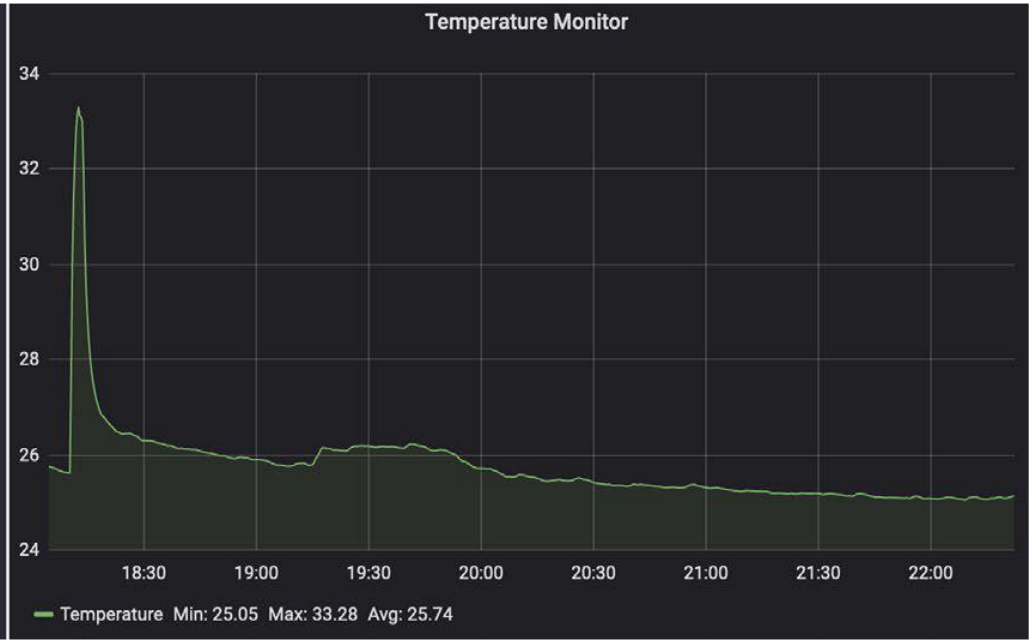
マルウェアが
EWS上で実行される

↓

攻撃者による工場への侵入、
マルウェアによる被害

# 産業用IoTデバイス向けのライブラリを改ざん：
# IoTデバイス開発ではオープンソースライブラリが活用される

```
void loop() {
#ifndef SOLO
  if (eth_on)
    ether.packetLoop(ether.packetReceive());
#endif

  long now = millis();
  if (now - lastMsgTime > SAMPLING_DELAY) {
    lastMsgTime = now;

    // Reading BME280 sensor data
    bme.takeForcedMeasurement(); // has no effect in normal mode
    temperature = bme.readTemperature();

    if (isnan(humidity) || isnan(temperature)) {
      Serial.println("BME280 reading issues");
      return;
    }

    char str[6];
    dtostrf(temperature, 4, 2, str);
    Serial.println(str);

#ifndef SOLO
    if (eth_on) // Send temperature readings
      ether.sendUdp(str, sizeof(str), srcPort, srv, dstPort);
#endif
  }
}
```

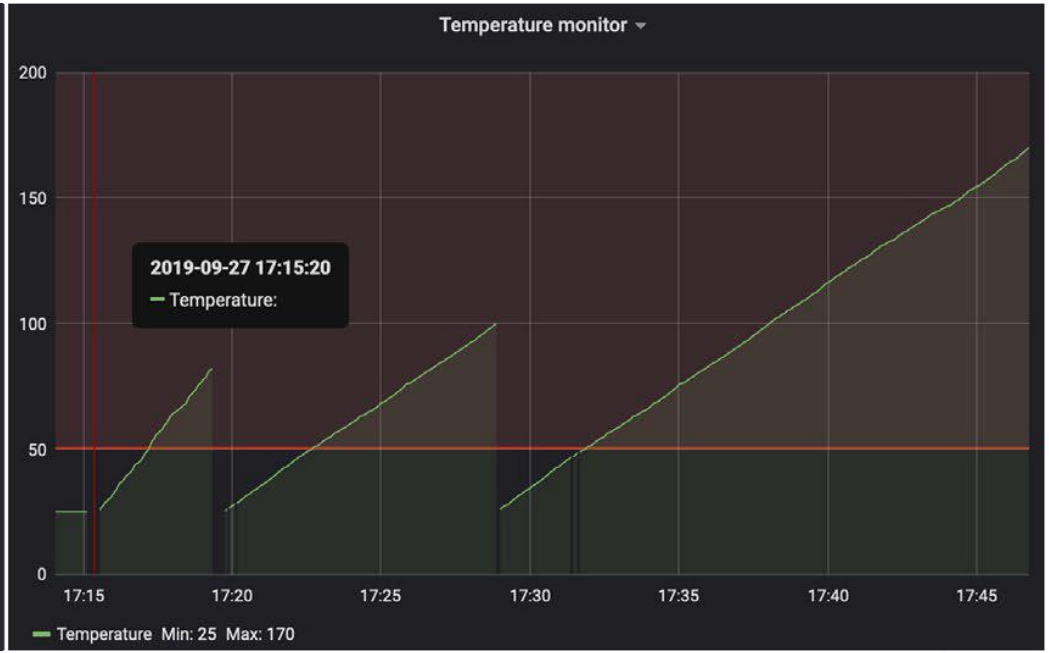



■画面：今回利用した温度センサーのファームウェア（左）と正常稼働時の温度表示（右）

# 産業用IoTデバイス向けのライブラリを改ざん：
# 温度センサーが正常に作動せず、製造ラインが止まる事態に

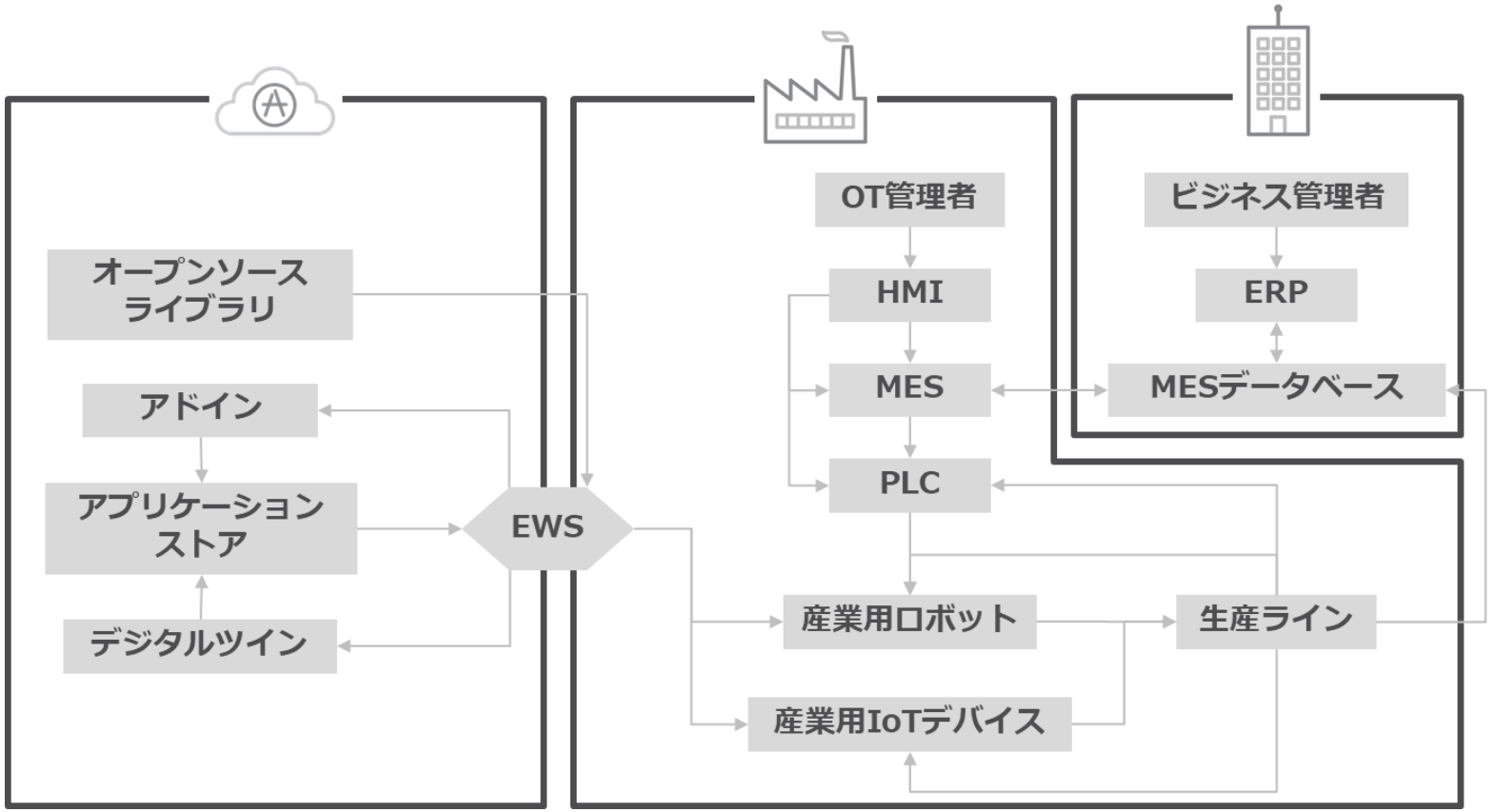
```
void loop() {
#ifndef SOLO
  if (eth_on)
    ether.packetLoop(ether.packetReceive());
#endif

  long now = millis();
  if (now - lastMsgTime > SAMPLING_DELAY) {
    lastMsgTime = now;

    // Reading BME280 sensor data
                                    as no effect in normal mode
    temperature = bme.readTemperature();

    if (isnan(humidity) || isnan(temperature)) {
      Serial.println("BME280 reading issues");
      return;
#ifndef SOLO
    if (eth_on) // Send temperature readings
      ether.sendUdp(str, sizeof(str), srcPort, srv, dstPort);
#endif
  }
}
```

見た目は変わらないが
中身が改ざんされている

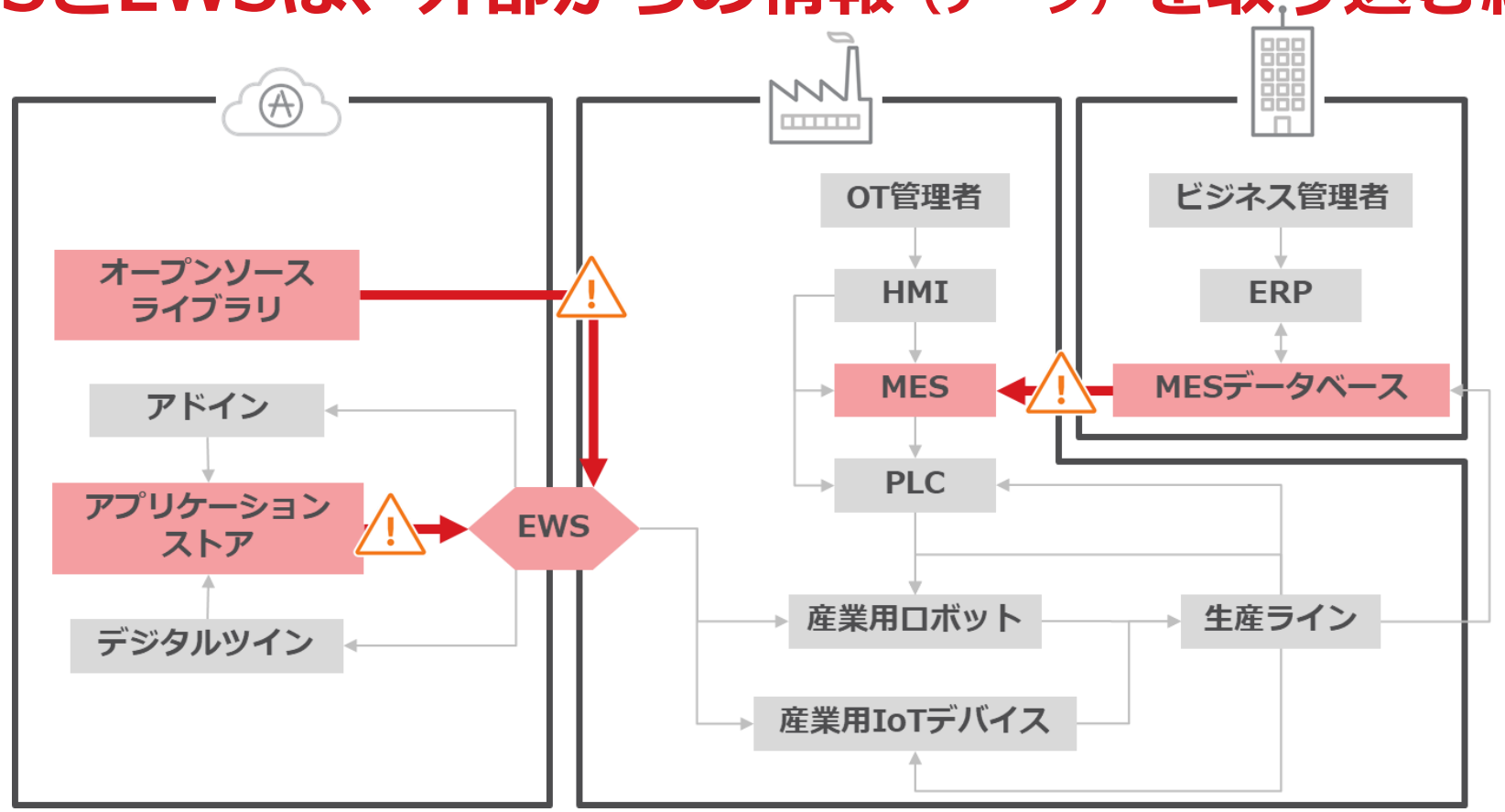



■画面：ライブラリの中身が改ざんされたことにより（左）、正常に温度が測れなくなった状態（右）

# スマート工場は工場以外の場所とデータ交換をする

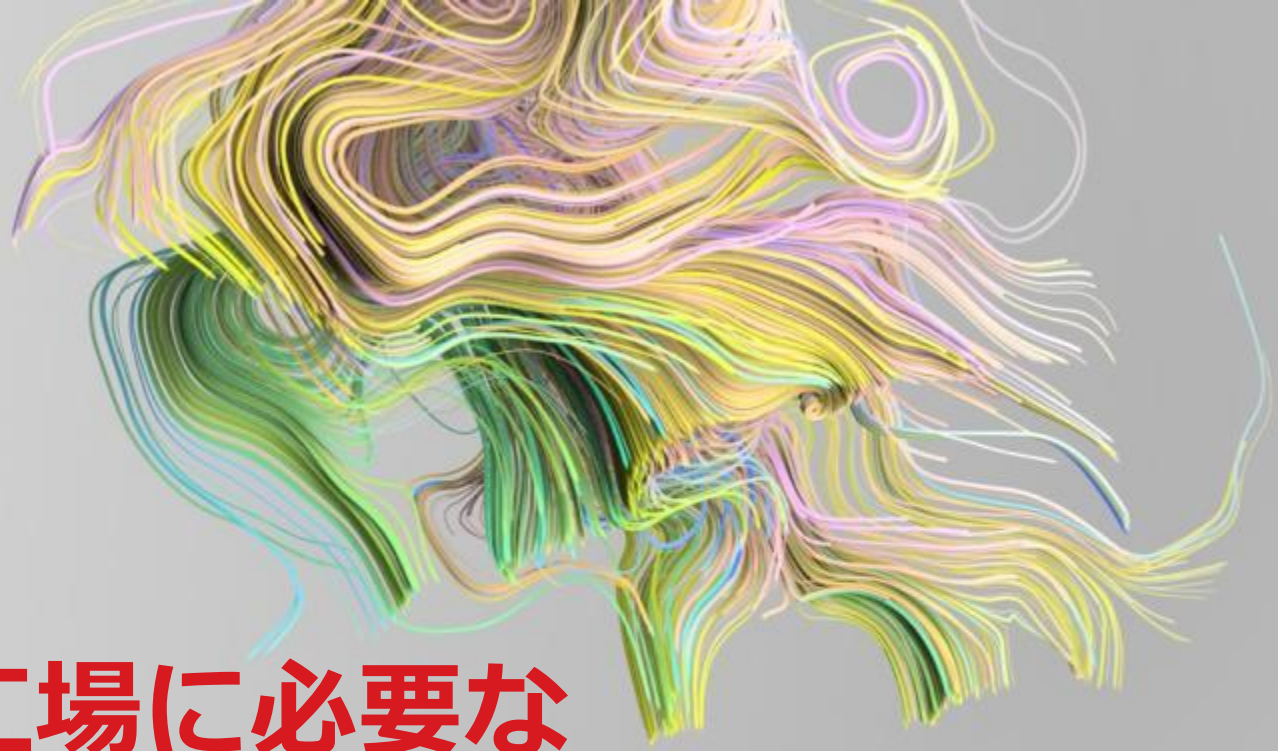


# MESとEWSは、外部からの情報（データ）を取り込む経路



# 実験結果まとめ

| 侵入経路 | 攻撃手法 | 被害 | セキュリティ対策 | セキュリティオーナー |
|---|---|---|---|---|
| ①アプリケーションストア →EWS | ■ 拡張ソフトを装ったマルウェアの流し込み | ■ マルウェアの侵入<br>■ データ窃取 | ■ サービスのセキュリティ強化<br>■ EWS保護<br>■ 開発ポリシー強化 | ■ 社内開発者<br>■ 産業機器メーカ |
| ②ITシステム →MES | ■ データベース値の改ざん | ■ 不良品の製造 | ■ データベース保護<br>■ アクセス制御 | ■ ITシステム管理者<br>■ OTシステム管理者 |
| ③オープンソースライブラリ →EWS | ■ ライブラリの改ざん | ■ デバイスの誤作動<br>■ 生産の中断 | ■ サプライチェーン管理<br>■ 開発ポリシー強化 | ■ 社内開発者<br>■ システムインテグレータ<br>■ IoTデバイスメーカ |

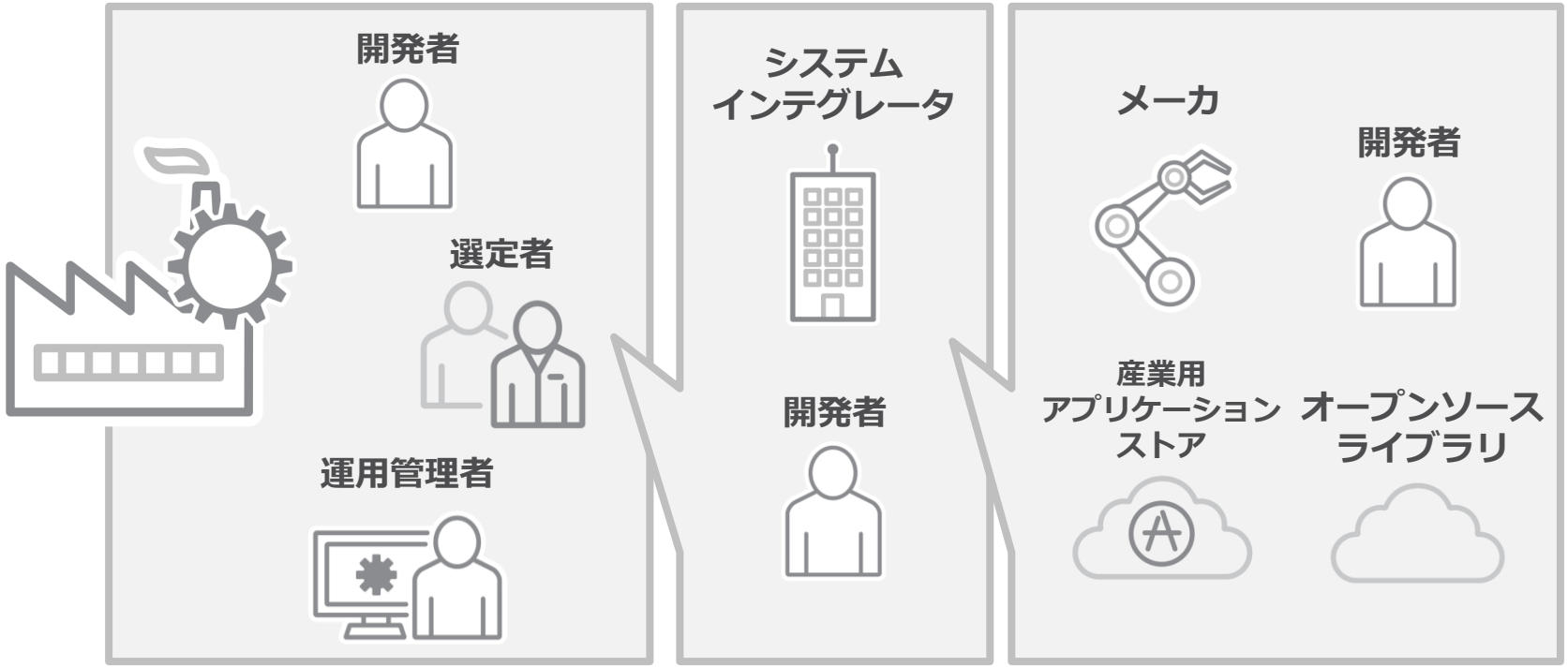


# スマート工場に必要なセキュリティ戦略

# スマート工場セキュリティには多くの人が関わる

| 侵入経路 | 攻撃手法 | 被害 | セキュリティ対策 | セキュリティオーナー |
|---|---|---|---|---|
| ①アプリケーションストア →EWS | ■ 拡張ソフトを装ったマルウェアの流し込み | ■ マルウェアの侵入<br>■ データ窃取 | ■ サービスのセキュリティ強化<br>■ EWS保護<br>■ 開発ポリシー強化 | ■ 社内開発者<br>■ 産業機器メーカ |
| ②ITシステム →MES | ■ データベース値の改ざん | ■ 不良品の製造 | ■ データベース保護<br>■ アクセス制御 | ■ ITシステム管理者<br>■ OTシステム管理者 |
| ③オープンソースライブラリ →EWS | ■ ライブラリの改ざん | ■ デバイスの誤作動<br>■ 生産の中断 | ■ サプライチェーン管理<br>■ 開発ポリシー強化 | ■ 社内開発者<br>■ システムインテグレータ<br>■ IoTデバイスメーカ |

# スマート化に伴いソフトウェアサプライチェーンは複雑さを増していく



# 実験結果からの洞察

スマート工場をより安全な場所にするには
**業界全体のセキュリティレベル**の向上が必須

**メーカ**
**インテグレータ**

**安全なものを作る**
**安全なものを提供する**
**（セキュリティバイデザイン）**

**ユーザ企業**

**あらゆる侵入経路を疑う・**
**侵入前提のセキュリティ対策**
**（ゼロトラスト）**

# セキュリティバイデザイン：自動化プログラムにも注目を

研究論文 “Rogue Automation（原題）”


ロボットの動きを定義する自動化プログラムはメーカ独自言語で書かれているが、セキュリティに課題

1. **開発プロセスの見直し**
   入力値チェック（バリデーション）が十全でなかったことでロボットの破損など予期せぬ物理被害の可能性。
2. **言語のプリミティブ機能に対するセキュリティ対策**
   ファイル読み書き、プログラム呼出/実行機能などのプリミティブな機能を持つ言語だが、下位リソースに対するアクセス制御がないためリソース乱用やマルウェア生成が可能。

**短・中・長期での業界全体の対策が必須**
https://jvn.jp/ta/JVNTA98870234/

# 有効なスマート工場のセキュリティを検討するには？：
# コンセプトに基づいたセキュリティ対策を

メーカ/インテグレータ

**セキュリティバイデザイン**

ユーザ

**ゼロトラスト**

コンセプトに基づく**セキュリティリスクの分析**

スマート工場のセキュリティの経営課題への**引き上げ・リソース付与**

業界標準規格（IEC62443など）を基準としたセキュリティ対策レベルの**共通認識**の構築

**セキュリティ対策の具現化**

**組織的対策**

**技術的対策**

# 最後に

メーカ/インテグレータ　ユーザの**双方**が

**セキュリティバイデザイン** と **ゼロトラスト** の考え方に基づき

**自覚的な協力関係**を構築することが
スマート工場セキュリティの鍵

TREND MICRO

# THE ART OF CYBERSECURITY

長年にわたりトレンドマイクロによって検出および阻止された未知の脅威。
**実際のデータを使用し、 アーティストの Brendan Dawes によって作成されました。**