改訂履歴：

- 2025年9月22日 例示ドメインの修正（P.33-P.35）
  なお資料改訂前の例示ドメインは、不適切に例示に使われたものであって実際には問題があったわけではない
- 2025年9月24日 例示IPアドレスの修正（P.23, P.33, P.34）
  なお資料改訂前の例示IPアドレス等は、不適切に例示に使われたものであって実際には問題があったわけではない



JPCERT CC®

# 制御システムにおけるDeception Systemと早期警戒網について

JPCERTコーディネーションセンター
制御システムセキュリティ対策グループ
阿部 真吾

# 目次

- 制御システムにおけるセキュリティの課題と名古屋工業大学との共同研究について
- Deception Systemについて
- 早期警戒網について
- 今後の課題

# 制御システムにおけるセキュリティの課題と名古屋工業大学との共同研究について

# 制御システムにおけるセキュリティの課題 | 1

- 情報システムへの攻撃と比べ、制御システムへの攻撃の検知手法はあまり確立されていない
  - 制御機器に残るログはシステム監視のデバッグやトラブルシュートが主目的でサイバー攻撃の調査は考慮されていない
  - アンチウイルスソフトが入っていない / 入れられない
  - そもそも現場は少人数運転によりセキュリティ対応のための人が確保できない

# 制御システムにおけるセキュリティの課題 | 2

- 制御システムに特化した情報を相互にやり取りし、共有するための仕組みがほとんどない
  - 多くの場合、発信されてくる情報を受け取って終わってしまう
  - 情報を受け取ってもその後のアクションに繋げづらい
  - そもそもどんな情報を相互に共有すると役に立つ（次のアクションに繋げられる）のかがよくわからない

# Deception Systemと早期警戒網のコンセプト

- 名古屋工業大学との共同研究として以下を検討している
- 現場担当者にあまり負担をかけることなく、攻撃を検知すると同時に防御するようなシステムはできないか（**Deception System：詳細は後述**）
  - 仮に攻撃されたとしても攻撃者に攻撃を断念させることで被害を抑える
- 検知した情報（脅威）を速やかに共有し、活用するためのシステムはできないか（**早期警戒網：詳細は後述**）
  - 早期にセキュリティの対応をできるようにすることで被害を抑える

# 名古屋工業大学の研究概要

1. 攻撃者に攻撃を断念させる防御方法の開発
   - 実際のネットワークと比べて巨大なネットワークに見せかけ、攻撃者を混乱させる

2. プラントの運転状態に合わせて通信をサイバー・タグアウト（攻撃経路の分離、攻撃範囲限定）（※）するための方法論の開発
   - 動的ゾーニングと経路制御によるタグアウト

   （※）一般にタグアウトとは機器の動力を遮断し、操作を禁止することを明示するためにタグを付け、可視化すること、この研究はタグアウトを電子的に行うことを想定

3. 機能実証用システムの開発
   - 1、2を組み合わせた**Deception System**
     - アセットオーナ側のネットワーク上に設置
   - **情報共有システム（早期警戒網）**
     - 各アセットオーナ、分析者で情報をやりとりするための基盤

**JPCERT/CCと共同研究**

# Deception Systemについて

# ICSネットワークにおけるサイバー脅威の発見

- 一般的なアプローチの例として以下が挙げられる
  - ICS機器に残るログの分析
    - システム監視のデバッグやトラブルシュートが主目的
      - サイバー攻撃の調査は考慮されていない
  - 正常動作からのアノマリ検知
    - ホワイトリストにより異常な動作を検知できる
    - 全ての異常を検知できるわけではない
      - ホワイトリスト登録時に異常な動作が混ざっていた場合
      - 正常な動作を悪用された場合
  - **ハニーポット**による検知
    - 脆弱なシステムを模倣することで攻撃者を誘導し、攻撃情報の収集・検知ができる

**ここに注目**

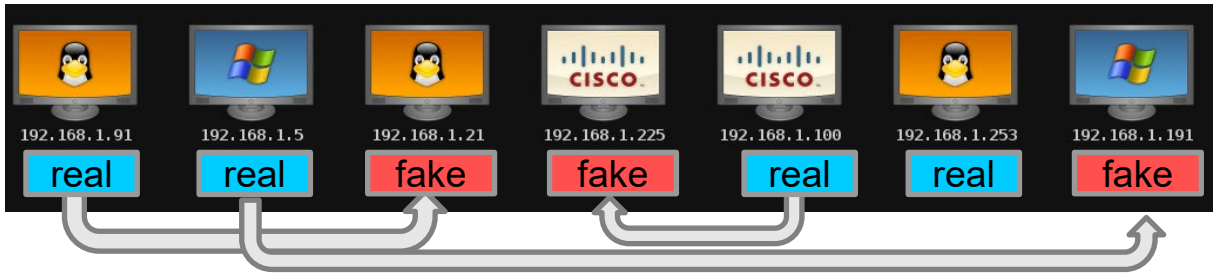
# Deceptionについて

- Deceptionとは
  - 騙す、欺く、欺まん
- サイバーにおけるDeception
  - 攻撃者を騙すことで・・・
    - 攻撃者の手法や情報を収集する
    - 実システムが攻撃されないように誘導する
    - 攻撃の邪魔をして時間を稼ぐ
  - 2016年にガートナーが発表した「注目すべき情報セキュリティ・テクノロジのトップ10」にも挙げられている

参考：https://www.sbbit.jp/article/cont1/32400

# Deception Systemに関する名古屋工業大学の研究｜1

- 攻撃者に攻撃を断念させる防御方法
  - 制御ネットワークに存在する機器のクローン（**ハニーポット**）を作成
  - 実際のネットワークと比べて巨大なネットワークに見せかけ、攻撃者を混乱させる
    - スキャンが来たらクローンのIPアドレスをランダムに変化させる



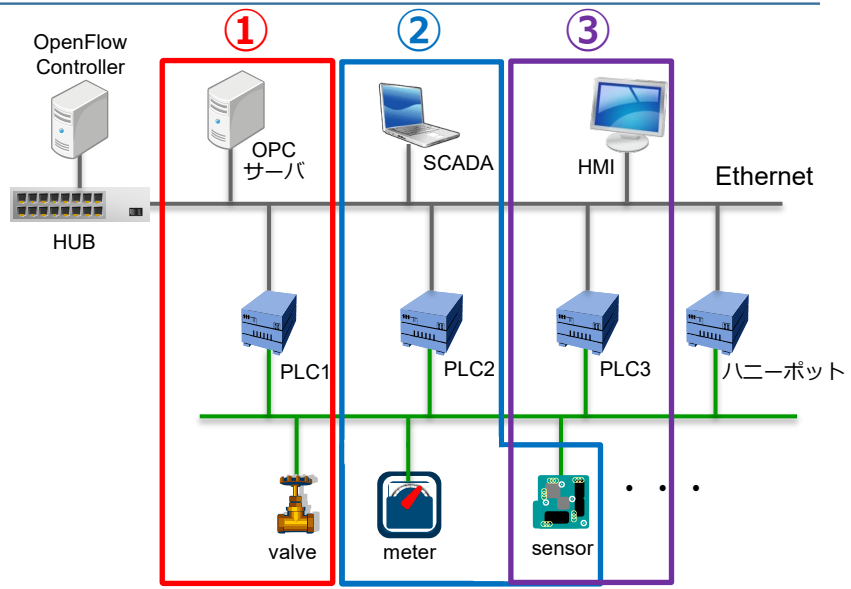

存在する機器から得られる情報を使ってクローンを作成する

# Deception Systemに関する名古屋工業大学の研究｜2

- プラントの運転状態に合わせて通信をサイバー・タグアウト（攻撃経路の分離、攻撃範囲限定）するための方法
  - 動的ゾーニングと経路制御によるタグアウト

（タグアウトの例）

- 最初は①のゾーンにある機器間のみ通信ができる（他の機器はタグアウト）
- この時、SCADAへの通信が発生したらそのままハニーポットに送る
- 経路の制御はHUBに繋がったOpenFlow Controllerにて行う




| Valve | Meter | Sensor | ・・・ |
|---|---|---|---|
| | | | |
| OPCサーバ | SCADA | PLC1 | ・・・ |
| | | | |

例示した状態を可視化したもの
（Webで確認できるシステムを用意）

# 一般的なハニーポット

- 到達したパケットをプロトコルやポート毎に振り分け、そのパケットの応答として設定されたアクションを返す
  - ハニーポットに構築したアプリケーションが応答する（高対話型）
  - アプリケーションの動作を模倣して応答する（低対話型）
    - 特定のパケットのみ応答するような実装（機器固有の情報を返すだけ）が一般的

- ハニーポットを活用して攻撃活動を観察する事例は既に存在している
  - 例）GasPot（海外のガソリンスタンドのガスタンクを管理するシステムを模倣して攻撃活動を観察した実験）

参考：https://www.blackhat.com/docs/us-15/materials/us-15-Wilhoit-The-Little-Pump-Gauge-That-Could-Attacks-Against-Gas-Pump-Monitoring-Systems-wp.pdf

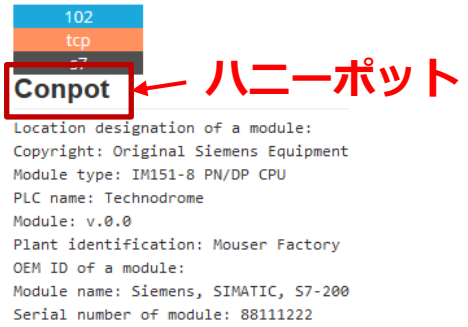
# 既存のハニーポットの課題｜1

Deception Systemとして利用する場合の課題を整理すると…

- インターネットに接続して構築することが前提
  - 外部から接続できる位置に置いて、攻撃者の行動や利用しているインフラ情報の一端を見ることを目的としている場合が多い
  - 内部ネットワークに置いて、横断的侵害を検知することを目的に利用しているケースはあまりない
    - 一部（主に情報系）で内部ネットワークの機器を模倣する欺まんシステムが出始めた程度

# 既存のハニーポットの課題　｜2

- 適切な応答をしないとハニーポットであることを見破られる可能性が高くなる
  - ハニーポット固有の情報が含まれていると、すぐに見破られてしまう
  - 本来変化するはずのものが常に同じ値だけを返すようになっていたり、不自然な値（シリアルナンバーがAll 0など）になっていても見破られる可能性が高い




抜粋： SHODAN
https://www.shodan.io/

# Traceback Honeypot System(THS)について

- Deception Systemとして利用するため、既存のハニーポットを拡張することを検討
  - イントラネットに設置することを前提とする
    - 内部侵入後の横断的侵害、感染拡大を検知することが目的
  - 攻撃者を騙す機能を高めるため、ある程度のコマンドに応答できるようにする
    - 長時間騙せるようにすることが目的
  - 攻撃元（＝感染源）の情報をプロファイルするため、攻撃元をスキャンする（カウンタースキャン）
    - 迅速なインシデント対応をすること、組織間で共有するための脅威情報を収集することが目的

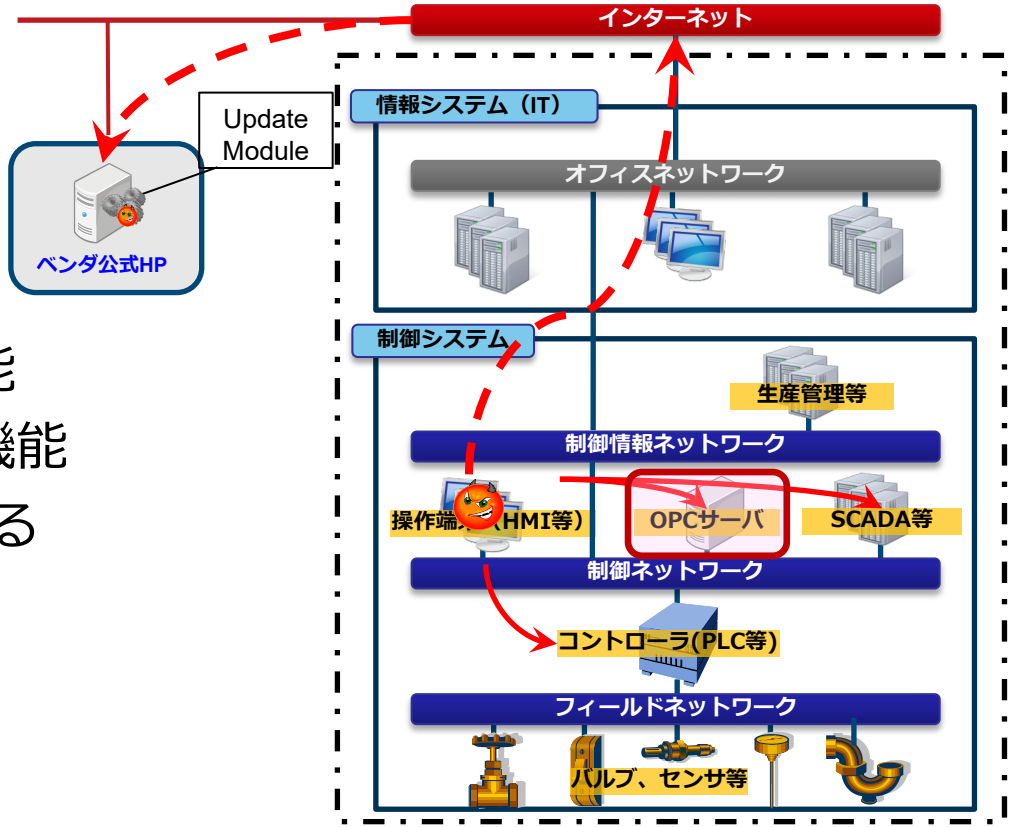
**Traceback Honeypot System(THS)と命名**

# THSでやりたいことの整理

- 横断的侵害、感染拡大を検知する
  - 想定する攻撃として以下を検討
    - 過去に海外で感染が確認されているHavex RAT
    - 海外の研究者が攻撃コンセプトとして発表した Modbus Stager、 PLC-Blaster
- 特定プロトコルのコマンドに対して、本物の機器と同じ（区別できない）応答を返す
  - レスポンスを比較して確認
- 攻撃元（＝感染源）にカウンタースキャンを行う
  - スキャンできることを確認

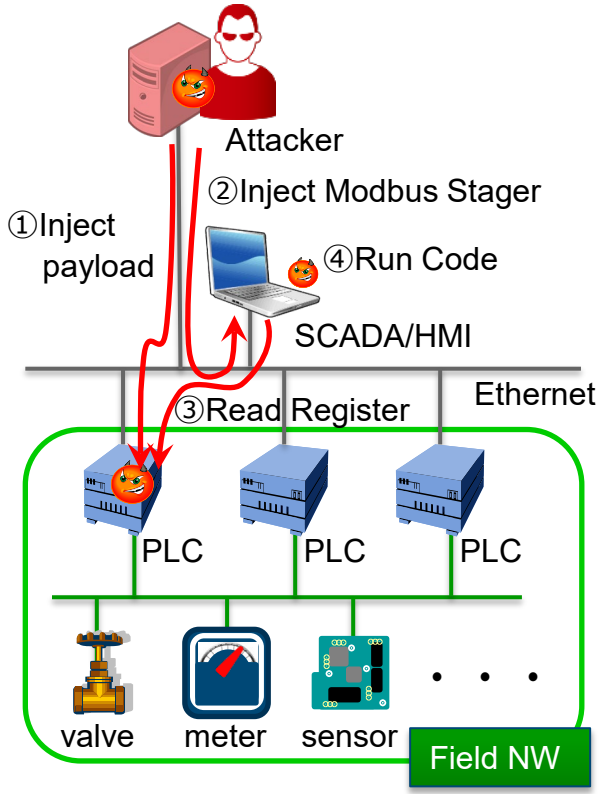
# 想定する攻撃（１）Havex RAT

- ICSベンダのWebサイトを改ざんし、ICSのソフトウエアにマルウエアを仕込む
- 以下の機能を持つ
  - OPCサーバスキャン機能
  - ネットワークスキャン機能
- 内部ネットワークに存在する正規のPCからのスキャンなので検知が難しい



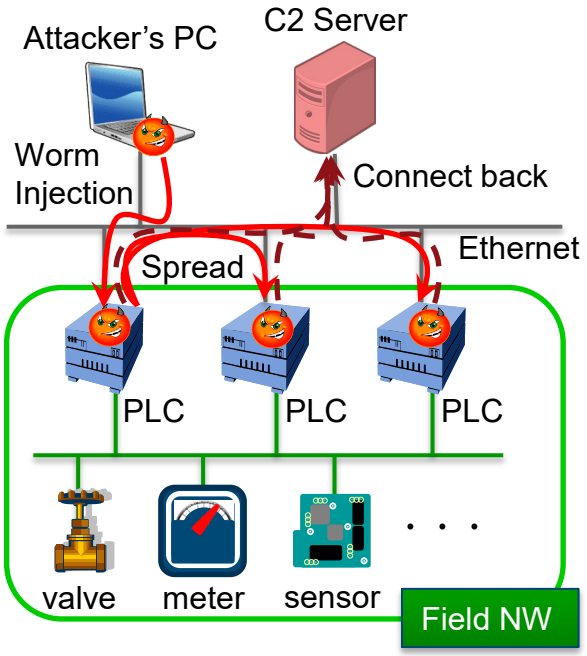
# 想定する攻撃（２）Modbus Stager（研究）

- Modbusプロトコルにより、PLCのHolding Registerを読み込むとSCADA/HMI等のPC側で不正なコマンドが実行される
- 事前に以下２つが必要
  - PLCのHolding Registerに不正なペイロードを書き込む
  - Modbus Stager（本体）をPCに感染させる
- 事前準備の後は内部ネットワークに閉じてModbusプロトコルにて不正なコマンドが実行されるため、検知が難しい



# 想定する攻撃（３） PLC-Blaster（研究）

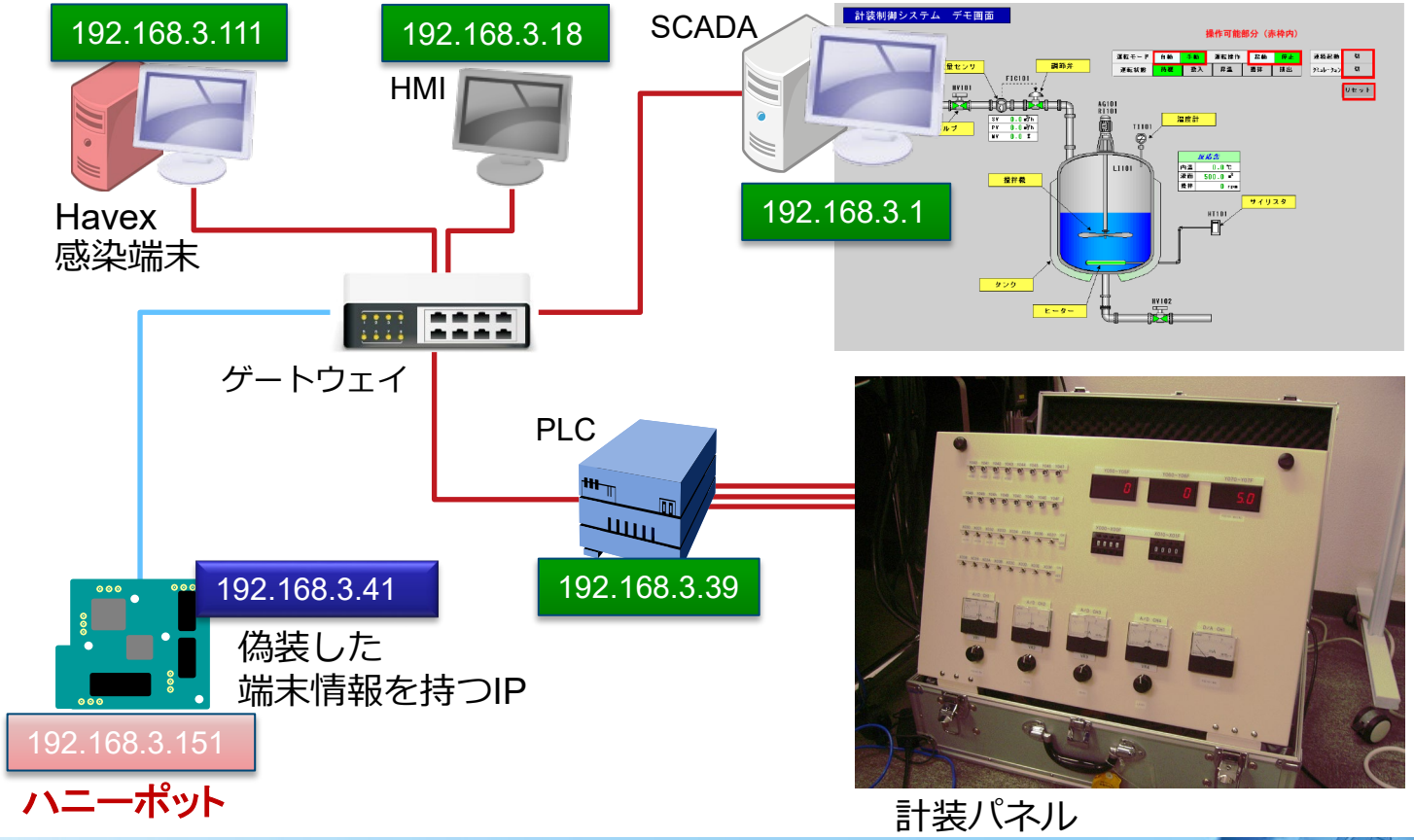
- ユーザプログラム領域を書き換え、正規プログラムの一部として入れ込む
- ネットワーク経由で1台に感染すると
  そこから同一ネットワーク内のPLCに感染を広げる
- 内部ネットワークに閉じて特定の制御プロトコルにて不正なプログラムが書き込まれるため、検知が難しい



# THSの動作検証の流れ

- ローカルネットワーク上にいくつかの制御機器とTHSを設置する
- 実際の制御機器からの応答パケットを使って偽装するための端末情報を作成する
- 攻撃元（＝感染源）から偽装している端末のIPアドレスに対して通信を行う
  - 今回はHavex RATとModbus Stagerで検証
- 攻撃元（＝感染源）に対してカウンタースキャンを行う

# ネットワーク構成



# 偽装する端末の設定情報の作成

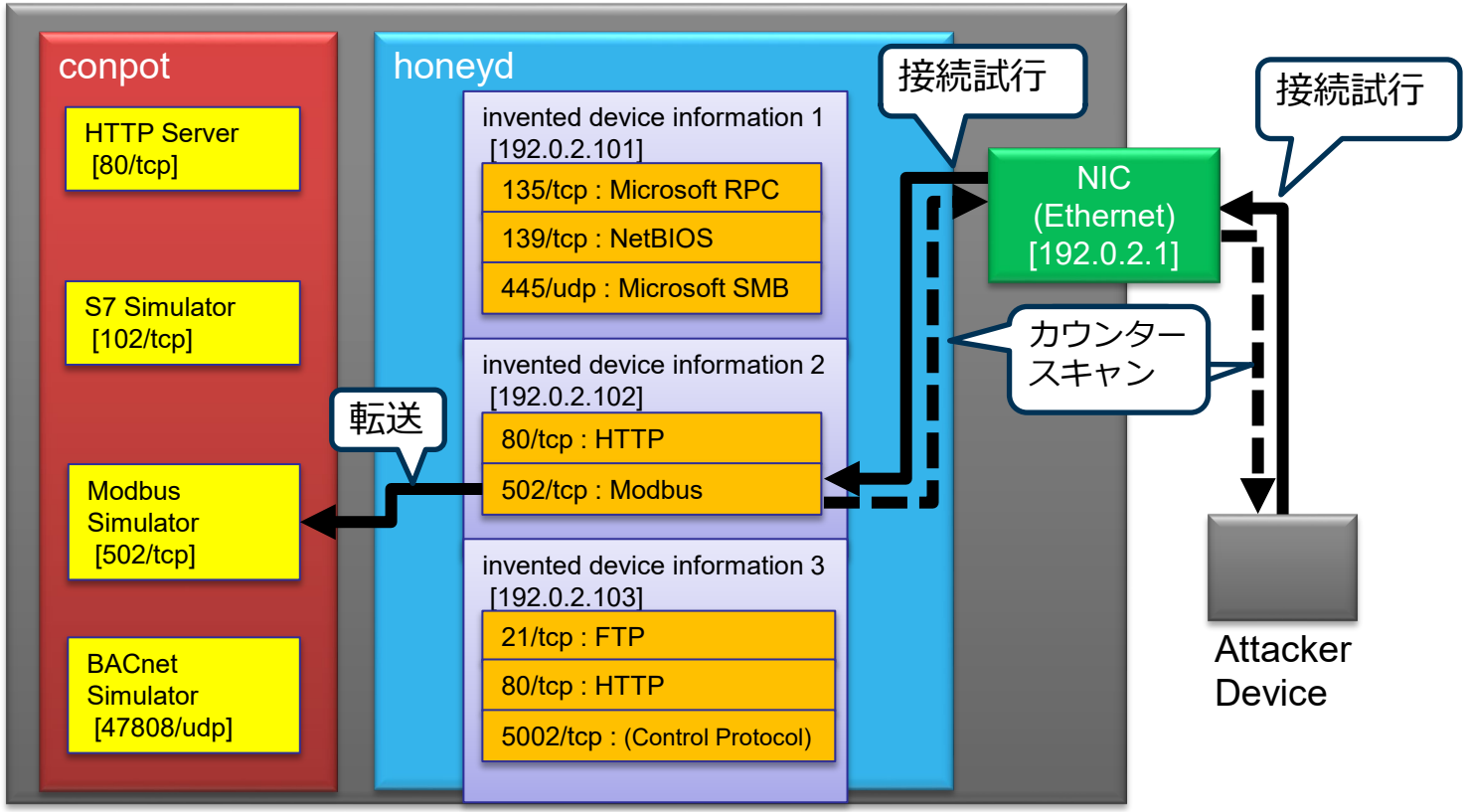
- 一般的なポートスキャンツールであるnmapで得られた応答をxml形式で保存する
- 保存したxmlをhoneyd（詳細は後述）のコンフィグに変換
  - この変換により任意のIPアドレスが振られた偽装マシンの設定が作成される

```
create machine1
set machine1 personality "Nomadix AG 5800 access gateway (VxWorks)“
set machine1 ethernet “AA:BB:CC:DD:EE“
add machine1 tcp port 102 ~
add machine1 tcp port 502 ~
~
set machine1 default tcp action closed
set machine1 default udp action closed
bind 192.168.3.41 machine1
```

192.168.3.0/24 内で現状存在しない値からランダムに選択振られた値
（偽装する対象(HMI)の端末のIPアドレスの第4オクテットは18）

# ソフトウエア構成

ハニーポット (Raspberry Pi)



# ソフトウエアの紹介

- Raspberry Pi
  - Linuxが動作するシングルボードコンピュータ
  - この上にTHSを構築
- honeyd
  - バーチャルホストが作れるハニーポットアプリケーション
  - （重複しない範囲で）任意のIPアドレスで特定のアプリケーションが動作しているように見せかけることができる
- conpot
  - 制御システム関連のプロトコルをエミュレートできるハニーポットアプリケーション

# ハニーポットに残るログの例

- Havexのスキャナがスキャンした様子がログに残る

Wiresharkでみた通信パケット

| No. | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|
| 13 | 192.168.3.111 | 192.168.3.151 | TCP | 66 | 49170→44818 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 16 | 192.168.3.151 | 192.168.3.111 | TCP | 60 | 44818→49170 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 |
| 17 | 192.168.3.111 | 192.168.3.151 | TCP | 54 | 49170→44818 [ACK] Seq=1 Ack=1 Win=65392 Len=0 |
| 18 | 192.168.3.111 | 192.168.3.151 | TCP | 66 | 49172→502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 19 | 192.168.3.151 | 192.168.3.111 | TCP | 60 | 502→49172 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 |
| 20 | 192.168.3.111 | 192.168.3.151 | TCP | 54 | 49172→502 [ACK] Seq=1 Ack=1 Win=65392 Len=0 |
| 21 | 192.168.3.111 | 192.168.3.151 | TCP | 66 | 49173→102 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 22 | 192.168.3.151 | 192.168.3.111 | TCP | 60 | 102→49173 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 |
| 23 | 192.168.3.111 | 192.168.3.151 | TCP | 54 | 49173→102 [ACK] Seq=1 Ack=1 Win=65392 Len=0 |
| 24 | 192.168.3.111 | 192.168.3.151 | TCP | 66 | 49174→11234 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 25 | 192.168.3.151 | 192.168.3.111 | TCP | 60 | 11234→49174 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 |
| 26 | 192.168.3.111 | 192.168.3.151 | TCP | 54 | 49174→11234 [ACK] Seq=1 Ack=1 Win=65392 Len=0 |
| 27 | 192.168.3.111 | 192.168.3.151 | TCP | 66 | 49175→12401 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 28 | 192.168.3.151 | 192.168.3.111 | TCP | 60 | 12401→49175 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 |

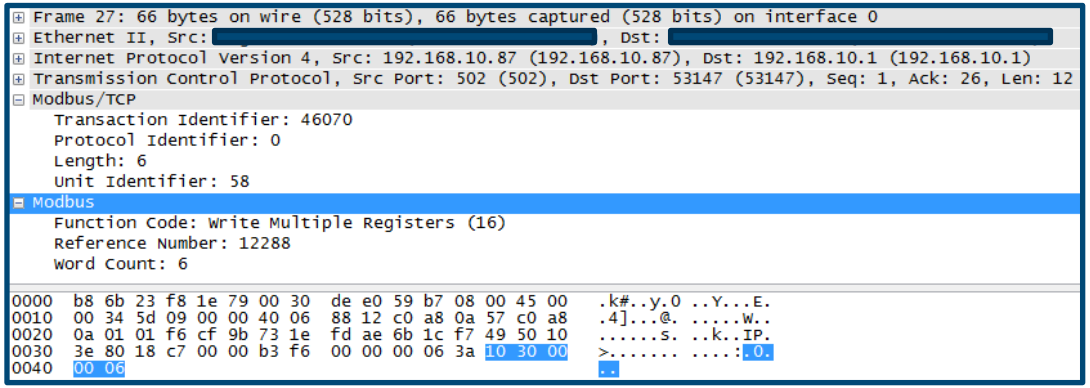
```
2017-03-27-21:53:10.1007 honeyd log started ------
2017-03-27-21:53:58.5141 tcp(6) S 192.168.3.111 49196 192.168.3.151 44818 [Windows 2000 RFC1323]
2017-03-27-21:53:58.5468 tcp(6) S 192.168.3.111 49198 192.168.3.151 502 [Windows 2000 RFC1323]
2017-03-27-21:53:58.5767 tcp(6) S 192.168.3.111 49199 192.168.3.151 102 [Windows 2000 RFC1323]
2017-03-27-21:53:58.6067 tcp(6) S 192.168.3.111 49200 192.168.3.151 11234 [Windows 2000 RFC1323]
2017-03-27-21:53:58.6366 tcp(6) S 192.168.3.111 49201 192.168.3.151 12401 [Windows 2000 RFC1323]
 :
```

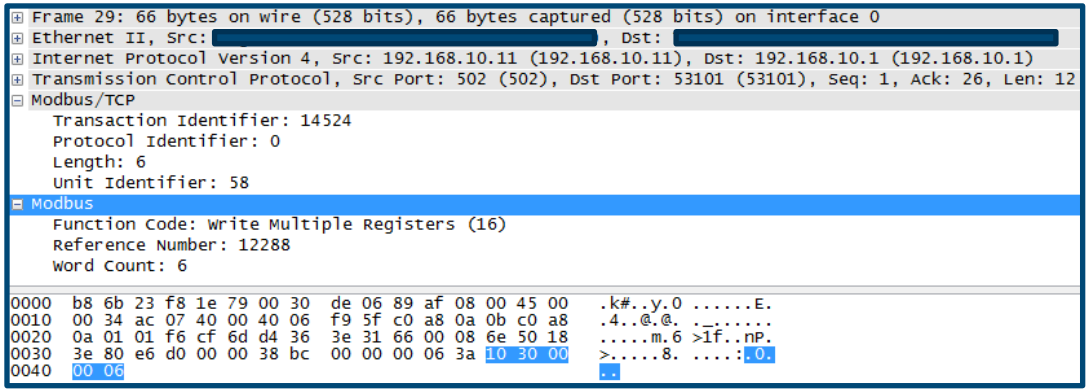
honeyd logfile

（※）Wiresharkはネットワークアナライザと呼ばれるアプリケーションで、
ネットワークに流れるパケットをキャプチャして見やすく表示することができる

# レスポンスの比較

- Wiresharkでパケットを比較しても全く変わらない




THSからのレスポンス

＝＝

実際の機器からのレスポンス

# カウンタースキャンした結果

```
$ sudo nmap -O 192.168.3.111

Starting Nmap 6.47 ( http://nmap.org ) at 2017-03-27 21:53 JST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try
using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.3.111
Host is up (0.00045s latency).
Not shown: 990 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
5357/tcp  open  wsdapi
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49157/tcp open  unknown
MAC Address: 00:0C:29:7C:F0:6C (VMware)
Device type: general purpose
Running: Microsoft Windows 2008|7
OS CPE: cpe:/o:microsoft:windows_server_2008::sp2 cpe:/o:microsoft:windows_7::-
cpe:/o:microsoft:windows_7::sp1 cpe:/o:microsoft:windows_8
OS details: Microsoft Windows Server 2008 SP2, Microsoft Windows 7 SP0 - SP1, Windows
Server 2008 SP1, or Windows 8
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 72.44 seconds
```

スキャンしてきた端末に対して
自動的にnmapを実行する

オープンしているポート、
MACアドレス、OSなどの
情報が得られる

# THSを動作検証した結果

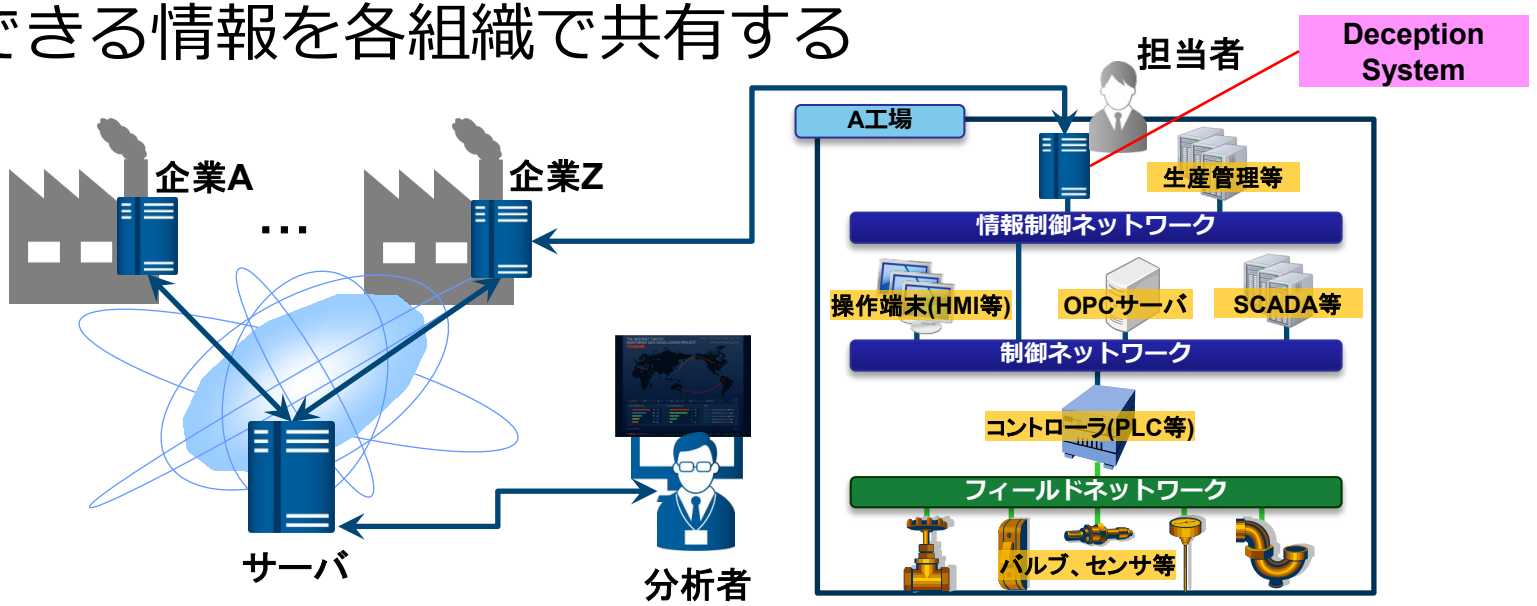
- 平時は動作しないため、ネットワークに負荷をかけない
- 特定のコマンドについては実際の端末と変わらないレスポンスを返すことができる
- 攻撃元（＝感染源）をスキャンしたとしても攻撃者に気づかれる可能性は低い
  - 攻撃中に乗っ取っている端末の通信監視をするとは考えづらいため

- 検知できる／できない攻撃が存在
  - 検知できる：ネットワークにパケットが発生する攻撃
    - ネットワークをスキャンするもの
    - ワームなどのようにネットワーク上の機器に対して感染を拡大しようとするもの
  - 検知できない：感染端末内で閉じている攻撃
    - キーロガーのようにキーボードからの入力を記録するようなもの
    - 感染端末のデータを破壊するもの

# 早期警戒網について

# 早期警戒網とは

- 各組織にDeception Systemを置き、異常の検知・攻撃のタグアウトを行う
- 検知した情報を分析し、**インディケータ**として活用できる情報を各組織で共有する



# インディケータとは

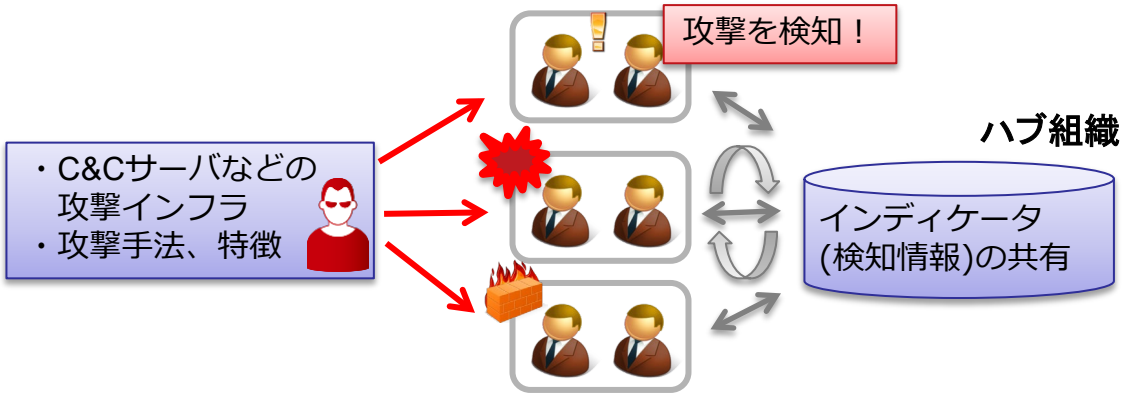
- 攻撃、または、攻撃の準備活動を選り分けるためのデータ・情報
  - 送信元・通信先のホスト名・IPアドレスやURL、ポート番号
  - ファイルの名前やハッシュ値、通信の特徴
  - 攻撃の手法や攻撃者の挙動

- インディケータを共有するための仕組み
  - 攻撃を受けた組織からの提供に基づき、ハブ組織（JPCERT/CC等）が各組織に通知

- インディケータを受け取った各組織における対応
  - 過去・現在・将来において攻撃の有無を調査
    - 可能な範囲で結果をハブ組織に提供

# インディケータ共有のイメージ

- ハブ組織がインディケータや攻撃手法などの知見を集約し、各組織に共有する事で、攻撃者のコスト増加（攻撃活動の手間を増やす、不正アクセスをさせないようにする）を図る
  - 組織間のインディケータなどを共有する
  - 各組織においてインディケータによる検知を行う
- ただし、インディケータ受領組織における対応コスト（検知情報とログの突き合わせなど）を勘案する必要性がある



# インディケータのイメージと活用方法｜1

- 推奨するログの確認期間
  - 20XX年 XX月以降

**ログ確認期間の絞り込みに活用**
**(大体1～3か月)**

- 攻撃に使用された通信先
  - example.com
    80/TCP(HTTP)
- 攻撃に使用された通信先 URL
  - http://example.com/hoge
    .php
- 攻撃に使用された通信先
  IP アドレス
  - 192.0.2.1 (国名) HTTPS

(攻撃の特徴例)

- 正規の Web サイトが改ざんされ、攻撃者に使用されている可能性がある

【注意】記載のドメイン等の情報は例示のために表示しているもので、実際には問題があったものではない

# インディケータのイメージと活用方法｜2

■ 改ざんされた Web サイト
  ― http://example.com/home, 192.0.2.1 (国名)

■ 改ざんされた Web サイトにアクセスした際の誘導先 URL
  ― http://example.net, 198.51.100.1

【注意】記載のドメイン等の情報は例示のために表示しているもので、実際には問題があったものではない

(攻撃の特徴例)

■ 以下のアプリケーションの脆弱性を悪用すると思われる
  ― 製品名 (CVE-20XX-0XXX)

↖ **端末のパッチ適用状況の確認に活用**

■ 誘導先URLは頻繁に変わるので誘導先URLでのブロックは難しい

■ 脆弱性が悪用された場合の誘導先へのアクセス回数
  ― n回 ↖ **ログ内のアクセス回数の確認に活用**

# インディケータのイメージと活用方法｜3

- ダウンローダの通信先
  - http://example.com/index.html
- HTTP ボットの通信先
  - http://example.net/id=[ランダムな文字列]

**活用できる情報は情報系と大きく変わらない**
**それをどうやって収集するかがポイント**

**Deception Systemを活用して**
**組織内の情報を収集**

(攻撃の特徴例)

- 以下のファイルがダウンロードされる **端末内の存在有無の確認に活用**
  - mimikatz.exe
  - Notepad.exe
    - サイズ：XXXXX bytes
    - ハッシュ値: MD5：aabbcc11223344556677889900ddeeff
    - SHA1：1234aabbcc11223344556677889900ddeeff4321

【注意】記載のドメイン等の情報は例示のために表示しているもので、実際には問題があったものではない

# Deception Systemで取得できる情報

- 現時点で取得できる情報を赤文字で記載
  - 通信ログ
  - 送信されるペイロード
  - OSの情報
    - バージョン、イベントログ、パッチ
  - アプリケーションの情報
    - バージョン、ログ、オープンしているポート
  - ファイル名　etc

- 攻撃元（＝感染源）にログインすれば赤文字以外の情報もDeception Systemで取得できる
  - ただしその場合、 Deception Systemにどのように認証情報を持たせるのかは検討が必要

# 情報共有する際のフォーマットについて

- 発信元によって記載フォーマットは異なる
  - 一般的なフリーテキストの形式だと情報の構造などに統一性がなく、自動的に処理したり、機械的に抽出することが難しい
  - 情報量が多くなると、人が目視し、手動で処理するなどの運用が難しい
  - **現場担当者の負担を減らす**ためにも、ある程度**処理を自動化**することを考慮したフォーマットが望ましい

- 標準フォーマットを使うと
  - 情報の構造や形式が統一されることで機械的に処理しやすい
  - 抽出した情報から検索するなどの処理の自動化が容易になる

# 既存の脅威・脆弱性情報共有の標準仕様

## ■標準仕様の目的

―共通仕様によって情報を齟齬なく共有する

検知に有効なサイバー攻撃を特徴づける指標

脅威情報に含まれる情報のスコープや種類

**情報を体系化して統一的に表現**

- 事象の特定
- 脅威度の分析
- 情報の管理

## ■標準仕様の例

**STIXを簡略化して利用できないか検討中**

MITRE: CVE CWE CVSS

Mandiant: OpenIOC

OASIS: STIX CybOX TAXII

# （参考）STIXについて

- Structured Threat Information eXpression ”
  「脅威情報構造化記述形式」

- サイバー攻撃活動を記述するための統一的な記述方式の一つ
- MITRE が中心となり仕様策定を進めてきたが、現在は OASIS が引き継いでいる
- 現在公開されている最新は STIX 2.0
  - JSON形式で記述する
  - 前バージョンのSTIX 1.2はXML形式で記述する
- OpenIoC などの脅威情報共有技術仕様や Snort や YARA などのツールと連携する機能も提供している

# 早期警戒網を活用した情報共有の流れ（イメージ）｜1

- 検知からインディケータを上げるまで
  - Deception Systemで不審な情報を検知し、アラートを上げる
    - 現場担当者が内容を確認して対応する
  - 検知した情報を標準フォーマットに変換する
    - 情報を暗号化する
  - 早期警戒網にアップロードする
    - 分析者がサーバにて定期的に内容を確認

# 早期警戒網を活用した情報共有の流れ（イメージ）｜2

- インディケータの受信から対応まで
  - 早期警戒網から標準フォーマットで脅威情報を受信する
  - Deception Systemで自組織に関連する情報かどうかを判定する
  - 自組織にて対応が必要な場合はアラートを上げる
    - 現場担当者が内容を確認して対応する

**各組織で早期に警戒態勢に入るための一助とする**

# 今後の課題

# 今後の課題（Deception System）

- 応答性能を高める（バリエーションを増やす）
  - ICSでは、利用しているシステムやアプリケーション、機器、ネットワークの構成、設定などが組織ごとに異なっており、攻撃対象を限定することが難しい
  - 今回は第一歩として汎用的なプロトコルに限定して実装

- 攻撃元（＝感染源）の情報をより多く収集する
  - 例えば攻撃元（＝感染源）に侵入して機器上で動作するシステムやサービス、アプリケーションを詳細に調査する
    - その情報をもとに１つの感染源をつぶすだけでなく、対策（アップデートやアクセス制限など）を横展開することができる

# 今後の課題（早期警戒網）

- 共有プラットフォームを構築する
  - 調査研究の一環としてJPCERT/CC内にサーバを設置し、協力いただける組織と情報のやり取りができるようにする
- 共有する情報の標準フォーマットを検討する
  - STIXをベースになるべく単純なものを目指す
- 自組織に関連する情報かどうかを判定するための機能を実装する
  - あらかじめ判定に必要な情報をDeception Systemに持たせることで対応できそう

# さいごに

- 今回発表した内容は実験段階であり、まだまだ課題がたくさんあります

- 興味がある、一緒に検討したい／試しに置いてデータを取ってみたい、という方がいらっしゃいましたら、是非お声がけください！ご協力をお願いいたします！

  - **連絡先：icsr@jpcert.or.jp**

**制御システムのセキュリティ向上のため、**
**これからも頑張ります！**

# お問合せ、インシデント対応のご依頼は

## JPCERTコーディネーションセンター

- Email：pr@jpcert.or.jp
- https://www.jpcert.or.jp/

## インシデント報告

- Email：info@jpcert.or.jp
- https://www.jpcert.or.jp/form/

## 制御システムインシデントの報告

- Email：icsr-ir@jpcert.or.jp
- https://www.jpcert.or.jp/ics/ics-form.html

# ご静聴ありがとうございました