Implicit Interactions Analysis

A Wastewater Treatment System Case Study

Jason Jaskolka, Ph.D., P.Eng.

Department of Systems and Computer Engineering
Carleton University, Ottawa, ON, Canada

jason.jaskolka@carleton.ca

@JasonJaskolka

October 21, 2020

Acknowledgement & Disclaimer

Acknowledgement

This material is based upon work supported by the U.S. Department of Homeland Security under Grant Award Number, 2015-ST-061-CIRC01.

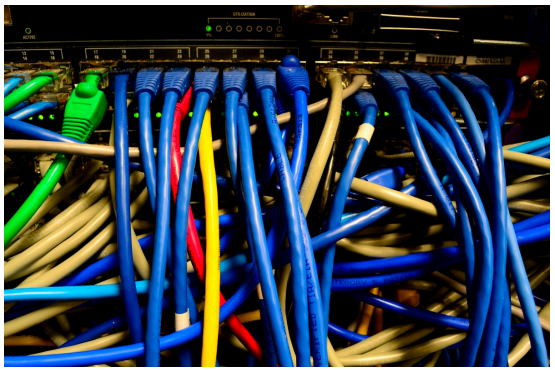
Disclaimer

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security.

Critical Infrastructure








Implicit Interactions

- Critical infrastructures consist of numerous components and even more interactions, some of which may be:
  - Unfamiliar, unplanned, or unexpected
  - Not visible or not immediately comprehensible

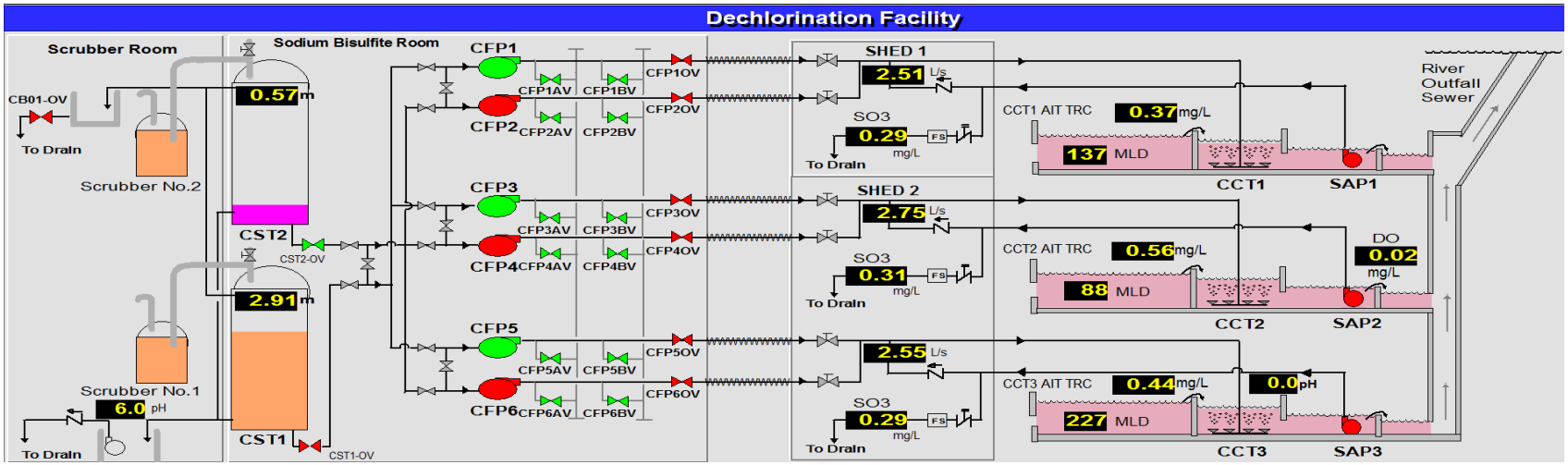
  } *Implicit Interactions*
- Can indicate unforeseen design flaws allowing for these interactions
- Constitute linkages of which designers are generally unaware
  $\Longrightarrow$ security vulnerability
- Can be exploited to mount cyber-attacks at a later time
  - Potential for unexpected system behaviours

Wastewater Treatment Facility

Wastewater Dechlorination Process

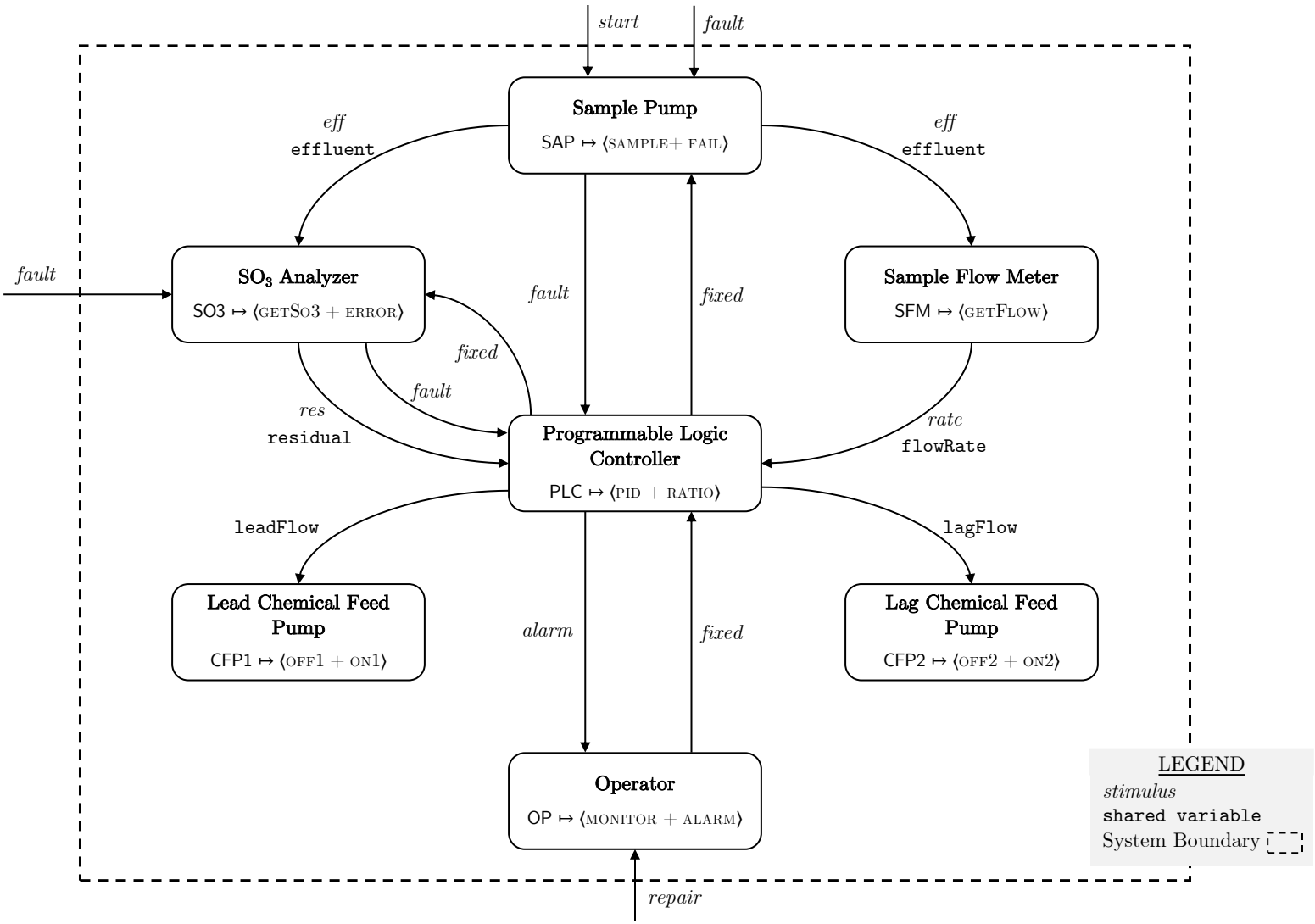
System Objective

Reduce the total residual chlorine in the plant's final effluent to comply with the Federal Government's regulated level



Provided by the SCADA system operators at a municipal wastewater treatment facility

Modeled System Operation



An Algebraic Modeling Framework

- **Communicating Concurrent Kleene Algebra (C^2KA)**
  - Formalism for system modeling
  - Expresses influence of stimuli on agent behaviour as well as communication through shared environments
- Three levels of specification
  1. Stimulus-Response Specification
  2. Abstract Behaviour Specification
  3. Concrete Behaviour Specification

Agent Specifications

○	*start*	*fault*	*eff*	*res*	*rate*	*off1*	*on1*	*off2*	*on2*	*alarm*	*fixed*	*repair*
PID	PID	RATIO	PID	PID	PID	PID	PID	PID	PID	PID	PID	PID
RATIO	RATIO	RATIO	RATIO	RATIO	RATIO	RATIO	RATIO	RATIO	RATIO	RATIO	PID	RATIO

λ	*start*	*fault*	*eff*	*res*	*rate*	*off1*	*on1*	*off2*	*on2*	*alarm*	*fixed*	*repair*
PID	n	*alarm*	n	n	n	n	n	n	n	n	n	n
RATIO	n	n	n	n	n	n	n	n	n	n	*fixed*	n

Table: Stimulus-response specification of Agent PLC

$$
\begin{aligned}
\text{SAP} &\mapsto \langle \text{SAMPLE} + \text{FAIL} \rangle \\
\text{SO3} &\mapsto \langle \text{GETSO3} + \text{ERROR} \rangle \\
\text{SFM} &\mapsto \langle \text{GETFLOW} \rangle \\
\text{PLC} &\mapsto \langle \text{PID} + \text{RATIO} \rangle \\
\text{CFP1} &\mapsto \langle \text{OFF1} + \text{ON1} \rangle \\
\text{CFP2} &\mapsto \langle \text{OFF2} + \text{ON2} \rangle \\
\text{OP} &\mapsto \langle \text{MONITOR} + \text{ALARM} \rangle
\end{aligned}
$$

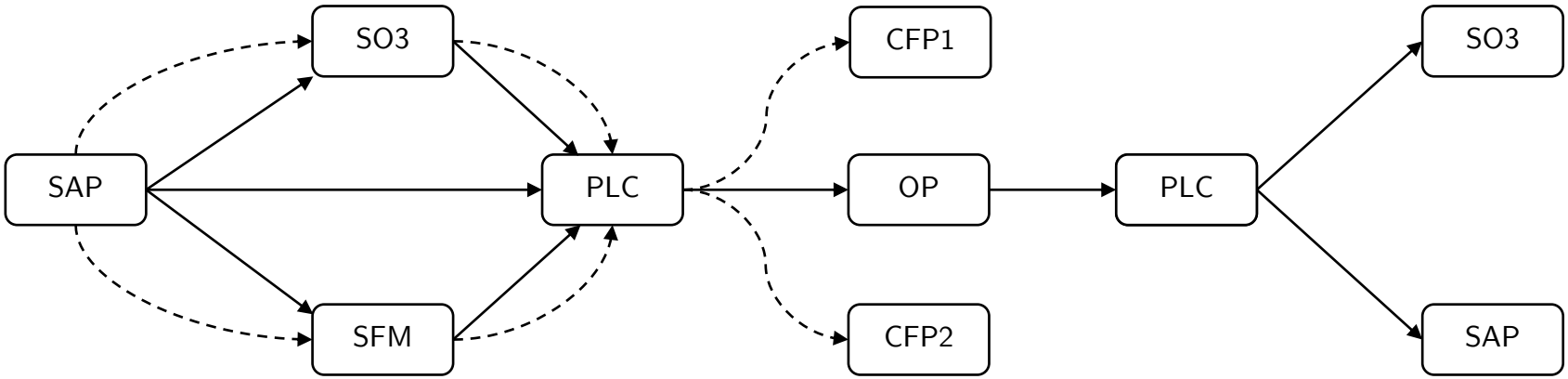
Figure: Abstract behavior specification for the system agents

Agent Specifications

$$
\begin{aligned}
\text{PLC} \mapsto \Big\{ \quad & \text{PID} \stackrel{\text{def}}{=} \text{if} \quad \texttt{flowRate} >= \texttt{FLOW\_SETPOINT} \longrightarrow \\
& \qquad \text{skip} \\
& \quad [] \quad \texttt{flowRate} < \texttt{FLOW\_SETPOINT} \longrightarrow \\
& \qquad \text{send } \textit{alarm} \\
& \quad \text{fi;} \\
& \quad \texttt{targetFlow} := \text{COMPUTE\_FLOW}(\texttt{residual}); \\
& \quad \text{if} \quad \texttt{targetFlow} > \texttt{MAX\_PUMP\_FLOW} \longrightarrow \\
& \qquad \text{send } \textit{on2}; \\
& \qquad \texttt{leadFlow} := \texttt{MAX\_PUMP\_FLOW}; \\
& \qquad \texttt{lagFlow} := \texttt{targetFlow} - \texttt{MAX\_PUMP\_FLOW} \\
& \quad [] \quad \texttt{targetFlow} \leq \texttt{MAX\_PUMP\_FLOW} \wedge \texttt{targetFlow} \geq \texttt{DEADBAND} \longrightarrow \\
& \qquad \texttt{leadFlow} := \texttt{targetFlow} \\
& \quad [] \quad \texttt{targetFlow} < \texttt{DEADBAND} \longrightarrow \\
& \qquad \text{send } \textit{off2}; \\
& \qquad \texttt{leadFlow} := \texttt{targetFlow} \\
& \quad \text{fi} \\
& \text{RATIO} \stackrel{\text{def}}{=} \text{skip} \quad \textit{// details not provided as part of the system description}
\end{aligned}
$$

Figure: Concrete behavior specification of Agent PLC

Intended System Interactions

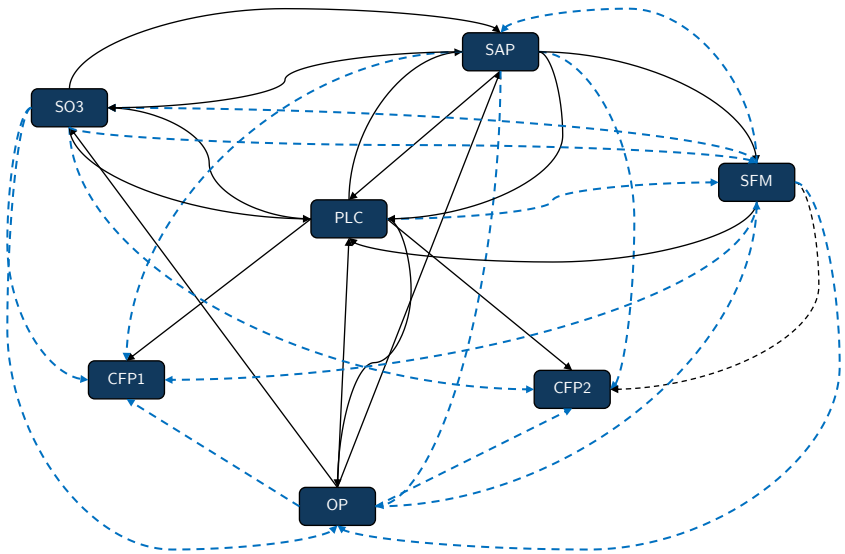


$\mathcal{P}_{\text{intended}}$ denotes the set of intended system interactions

Identifying Implicit Interactions

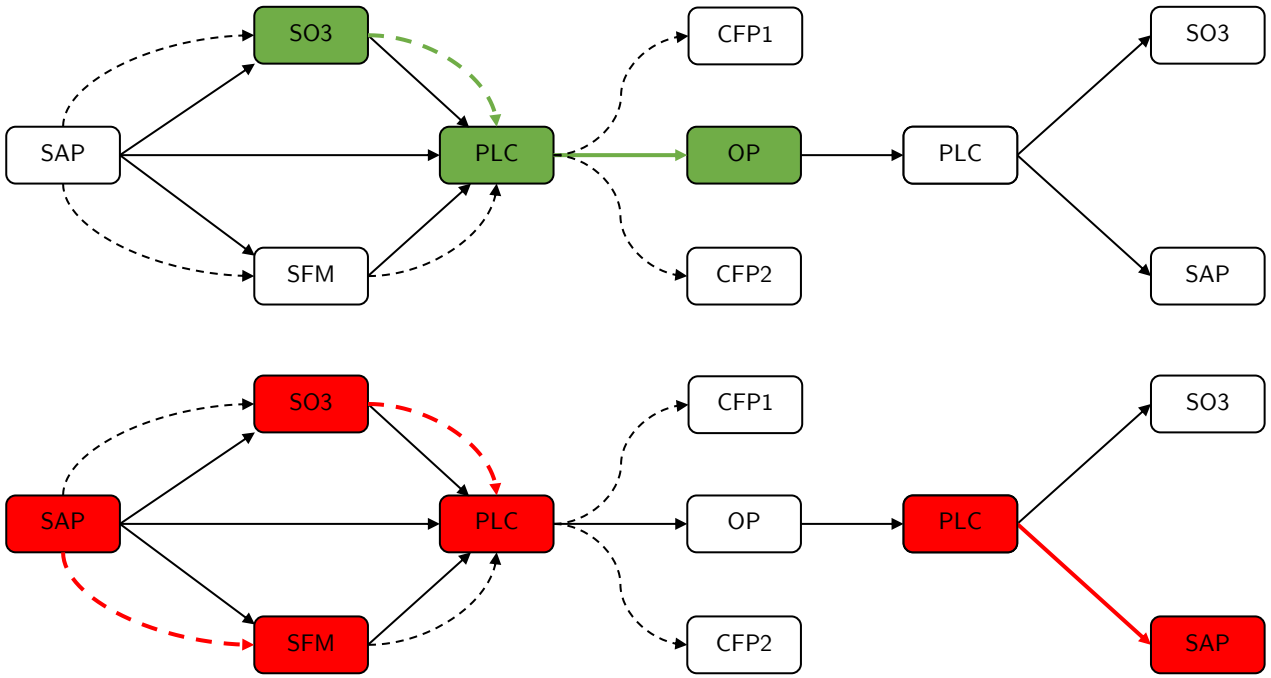
1. Determine the potential communication paths that exist from the system specification

```
$ pfc system agentPLC agentSAP
  P ~> S: True
    PLC  ->S  OP  ->S  SAP
    PLC  ->S  SAP
    PLC  ->S  OP  ->S  SO3  ->S  SAP
    PLC  ->S  SO3  ->S  SAP

$ pfc system agentSO3 agentSFM
  SO3 ~> SFM: True
    SO3 ->E PLC ->S OP ->S SAP ->E SFM
    SO3 ->S PLC ->S OP ->S SAP ->E SFM
    SO3 ->E PLC ->S SAP ->E SFM
    SO3 ->S PLC ->S SAP ->E SFM
    SO3 ->S SAP ->E SFM
```



Identifying Implicit Interactions

2. Determine if a potential communication path is an implicit interaction
   - *Example:* Consider the following potential communication paths: SO3 $\rightarrow_{\mathcal{E}}$ PLC $\rightarrow_{\mathcal{S}}$ OP and SO3 $\rightarrow_{\mathcal{E}}$ PLC $\rightarrow_{\mathcal{S}}$ SAP $\rightarrow_{\mathcal{E}}$ SFM



Severity Analysis

Definition (Severity Measure)

Let p be a possible interaction in a system with intended system interactions $\mathcal{P}_{\text{intended}}$. The *severity of* p is computed by:

$$\sigma(p) = 1 - \max_{q \in \mathcal{P}_{\text{intended}}} \left\{ \frac{|\text{lcs}(p, q)|}{|p|} \right\}$$

where $\text{lcs}(p, q)$ is the longest common substring of interactions p and q.

less overlap $\Longrightarrow$ higher severity $\Longrightarrow$ more unexpected

Exploitability Analysis

Definition (Exploitability Measure)

The *exploitability* of an implicit interaction $p_n^{\mathcal{T}_n}$ is computed recursively:

$$\xi\left(p_n^{\mathcal{T}_n}\right) = \begin{cases} \xi\left(p_{n-1}^{\mathcal{T}_{n-1}}\right) \dfrac{|\mathrm{Infl}(\mathsf{A}_{n-1}) \cap \mathrm{attack}\left(p_n^{\mathcal{T}_n}\right)|}{|\mathrm{Infl}(\mathsf{A}_{n-1})|} & \text{if} \quad \mathcal{T}_n = \mathcal{S} \ \wedge \ n > 1 \\ \xi\left(p_{n-1}^{\mathcal{T}_{n-1}}\right) \dfrac{|\mathrm{Ref}(\mathsf{A}_{n-1}) \cap \mathrm{attack}\left(p_n^{\mathcal{T}_n}\right)|}{|\mathrm{Ref}(\mathsf{A}_{n-1})|} & \text{if} \quad \mathcal{T}_n = \mathcal{E} \ \wedge \ n > 1 \\ 1 & \text{otherwise} \end{cases}$$

where for any agent $\mathsf{A} \in \mathcal{A}$

- $\mathrm{Infl}(\mathsf{A})$: set of stimuli that can influence the behavior of A
- $\mathrm{Ref}(\mathsf{A})$: set of referenced variables for A
- $\mathrm{attack}\left(p_n^{\mathcal{T}_n}\right)$: set of possible ways a compromised source of $p_n^{\mathcal{T}_n}$ can influence the behavior of the sink

higher exploitability $\Longrightarrow$ more ways to influence behaviours

Software Prototype

ImplicitInteractionsTool

```
IMPLICIT PATHS: SO3 ~>+ PLC
-----------------------
        SO3  ->S  SAP  ->S  PLC
        SO3  ->S  SAP  ->E  SFM  ->E  PLC

SO3 ~>+ SAP: True
-----------------------
ALL PATHS: SO3 ~>+ SAP
-----------------------
        SEVERITY = 0.33         SO3  ->E  PLC  ->S  OP  ->S  SAP
        SEVERITY = 0.33         SO3  ->S  PLC  ->S  OP  ->S  SAP
        SEVERITY = 0.50         SO3  ->E  PLC  ->S  SAP
        SEVERITY = 0.50         SO3  ->S  PLC  ->S  SAP
        SEVERITY = 1.00         SO3  ->S  SAP
-----------------------
IMPLICIT PATHS: SO3 ~>+ SAP
-----------------------
        SO3  ->E  PLC  ->S  OP  ->S  SAP
        SO3  ->S  PLC  ->S  OP  ->S  SAP
        SO3  ->E  PLC  ->S  SAP
        SO3  ->S  PLC  ->S  SAP
        SO3  ->S  SAP

SO3 ~>+ SFM: True
-----------------------
ALL PATHS: SO3 ~>+ SFM
-----------------------
```

Software Prototype: Sample Output

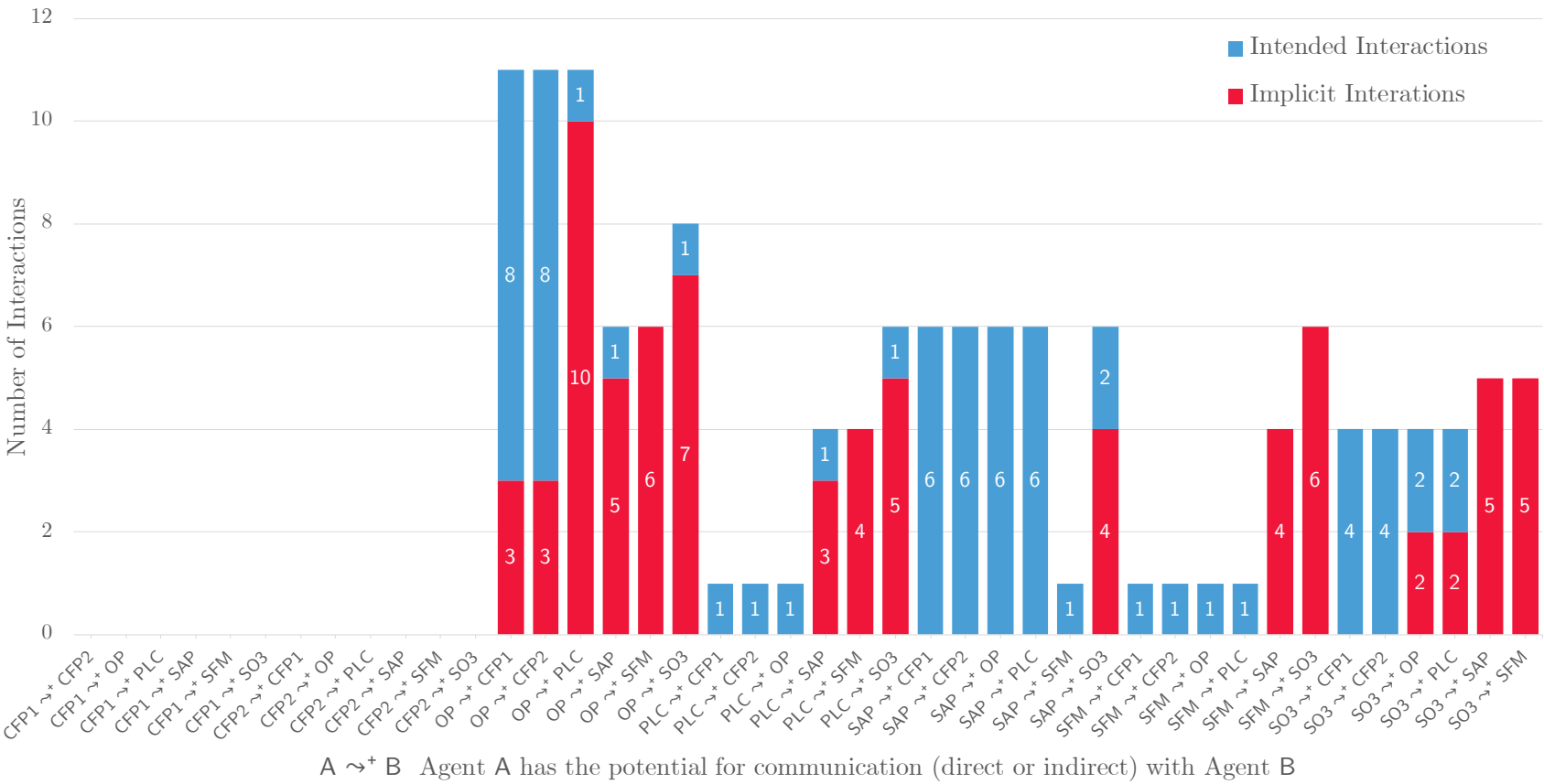
Identification & Severity

```
 ------------------------
 ALL PATHS: PLC ~>+ SAP
 ------------------------
  Severity = 0.50 PLC ->S OP ->S SAP
  Severity = 0.00 PLC ->S SAP
  Severity = 0.67 PLC ->S OP ->S SO3 ->S SAP
  Severity = 0.50 PLC ->S SO3 ->S SAP
 ------------------------
 IMPLICIT PATHS: PLC ~>+ SAP
 ------------------------
  PLC ->S OP ->S SAP
  PLC ->S OP ->S SO3 ->S SAP
  PLC ->S SO3 ->S SAP

------------------------
ALL PATHS: SO3 ~>+ SFM
------------------------
  Severity = 0.50 SO3 ->E PLC ->S OP ->S SAP ->E SFM
  Severity = 0.50 SO3 ->S PLC ->S OP ->S SAP ->E SFM
  Severity = 0.67 SO3 ->E PLC ->S SAP ->E SFM
  Severity = 0.67 SO3 ->S PLC ->S SAP ->E SFM
  Severity = 0.50 SO3 ->S SAP ->E SFM
------------------------
IMPLICIT PATHS: SO3 ~>+ SFM
------------------------
  SO3 ->E PLC ->S OP ->S SAP ->E SFM
  SO3 ->S PLC ->S OP ->S SAP ->E SFM
  SO3 ->E PLC ->S SAP ->E SFM
  SO3 ->S PLC ->S SAP ->E SFM
  SO3 ->S SAP ->E SFM
```

Attack Scenarios & Exploitability

```
Implicit Interaction =  PLC ->S OP ->S SAP
Attack Scenario = {alarm, repair}
Exploitability    = 1.0

Implicit Interaction =  PLC ->S OP ->S SO3 ->S SAP
Attack Scenario = {}
Exploitability    = 0.0

Implicit Interaction =  PLC ->S SO3 ->S SAP
Attack Scenario = {}
Exploitability    = 0.0

Implicit Interaction =  SO3 ->E PLC ->S OP ->S SAP ->E SFM
Attack Scenario = {flowrate, residual, targetFlow}
Exploitability    = 1.0

Implicit Interaction =  SO3 ->S PLC ->S OP ->S SAP ->E SFM
Attack Scenario = {fault}
Exploitability    = 0.5

Implicit Interaction =  SO3 ->E PLC ->S SAP ->E SFM
Attack Scenario = {}
Exploitability    = 0.0

Implicit Interaction =  SO3 ->S PLC ->S SAP ->E SFM
Attack Scenario = {fixed}
Exploitability    = 0.5

Implicit Interaction =  SO3 ->S SAP ->E SFM
Attack Scenario = {fault, fixed}
Exploitability    = 1.0
```

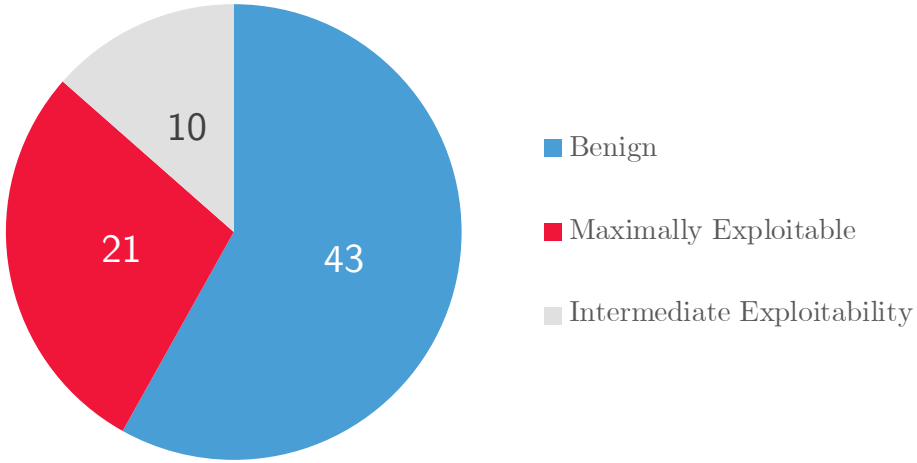
Experimental Results: Identification



A ⇝⁺ B Agent A has the potential for communication (direct or indirect) with Agent B

Experimental Results: Exploitability Analysis

- **74** of **141** interactions ($\approx 52\%$) are identified as *implicit interactions*



- Result of the potential for out-of-sequence messages or reads/writes from system agents
  - Due to cyber-attack or failure
- Demonstrates hidden complexity and coupling among agents
  - Potential for unexpected system behaviours

Model Validation

- Detailed reports of the specifications and analysis results were provided to SCADA operators
  1. Informal system description
  2. C^2KA system model specification
  3. System analysis results generated by the software prototype
- Reviewed, validated (by domain expert inspection), and approved by SCADA operators and Senior Control Systems Engineer
  - Confirmed that the system model and analysis results are valid in real-world contexts and scenarios

Domain Expert Questionnaire

- Distributed to relevant stakeholders at the municipal wastewater treatment facility that provided the case study system
- Consisted of two parts:
  1. *Part I*: Modeling and Analysis of the Dechlorination Process
  2. *Part II*: Approach for Identifying and Analyzing Implicit Interactions
- Completed by **6 respondents**, each of which were involved in SCADA operations

Questionnaire Results: Part I

1. Did the obtained and presented analysis results match your expectations based on your understanding of the Wastewater Dechlorination System?
   - **6 of 6** participants answered **Yes**
     - *"It exceeded our expectations because it provided us with an alternative perspective on the analysis of the dechlorination process."*
2. Are the obtained and presented analysis results understandable?
   - **6 of 6** participants answered **Yes**
3. Are the obtained and presented analysis results valuable to you, your team, and/or your organization/others?
   - **6 of 6** participants answered **Yes**
     - *"It highlights subtle weaknesses of certain interactions in the process."*

Questionnaire Results: Part II

4. Do you believe that the approach for identifying and analyzing implicit interactions has value?
   - **6 of 6** participants answered **Yes**
     - *"It identifies some weaknesses in the process."*
5. If you had a tool to perform the analysis offered by the approach for identifying and analyzing implicit interactions, would it benefit your activities?
   - **6 of 6** participants answered **No**
     - *"Such a tool should be used by the integrator or developer in the early stages of the design."*
6. If you had a tool to perform the analysis offered by the approach for identifying and analyzing implicit interactions, would you use it?
   - **6 of 6** participants answered **Maybe**
     - *"Such a tool could be used to verify the integrator's or developer's design."*

Questionnaire Results: Strengths

8. In your opinion, what are the strengths of the approach for identifying and analyzing implicit interactions?
   - *"Any system that highlights potential problems is helpful"*
   - *"The analysis is good at pointing to the source of problem areas/components in the system"*
   - *"The value of the approach is in finding issues early in the engineering design of systems; this is helpful for consultants, etc."*
   - *"The analysis may also find a use as part of the internal continuous improvement processes, especially, if it is easy to perform with good tool support"*

Questionnaire Results: Weaknesses

9. In your opinion, what are the weaknesses of the approach for identifying and analyzing implicit interactions?
   - *"It requires end-user expertise on the subject matter"*
   - *"The analysis may be more useful for system integrators rather than system operators; as operators, this kind of analysis would be nice to have included in proposal from integrators that are contracted to upgrade the system, etc."*
   - *"It would be nice if in additions to showing the implicit interactions, some advice on mitigations for the identified interactions could be provided"*
   - *"A summary of problematic areas would be helpful as part of the reporting of the results"*

Questionnaire Results: Other Feedback

10. Please provide any other comments/feedback about the approach for identifying and analyzing implicit interactions?
    - *"If used in the early stages of system development it can identify hidden problems and perhaps provide cost savings and time."*

Lessons Learned

1. Approaches are useful for identifying potential issues early in the design of the system
   - Promise for adoption and use among system integrators in support security assurance efforts
   - Can provide evidence that systems have been designed to be resilient to cyber-threats
2. Room for improvement with scalability and tool support
   - More effort to efficiently applying these approaches to conduct the analysis
   - Need to consider user-friendly tools to reduce end-user expertise requirements
3. Approaches can be applied in other contexts
   - Analogous communication and dependencies are found in nearly all industrial control systems

Conclusion

- Implicit interaction analysis provides a step towards uncovering potential cybersecurity vulnerabilities
  - Help to improve system stability, safety, and security
- Demonstrated real-world applicability of implicit interaction analysis
  - Enhanced understanding of the hidden complexity and coupling in the systems
  - Results can inform mitigation efforts at early stages of the system design, including prioritization
- Approaches and results were found to be valuable, understandable, and exceed expectations

Related Publications


J. Jaskolka
Identifying and Analyzing Implicit Interactions in a Wastewater Dechlorination System.
6th Workshop on the Security of Industrial Control Systems and of Cyber-Physical Systems, September 2020, (To Appear).


J. Jaskolka
Evaluating the Exploitability of Implicit Interactions in Distributed Systems.
arXiv:2006.06045 [cs.CR], June 2020.


J. Jaskolka and J. Villasenor.
An Approach for Identifying and Analyzing Implicit Interactions in Distributed Systems.
IEEE Transactions on Reliability, 66(2):529-546, June 2017.


J. Jaskolka and J. Villasenor.
Identifying Implicit Component Interactions in Distributed Cyber-Physical Systems.
Proceedings of HICSS-50, 5988–5997, January 2017.

Thank You

CyberSEA
Research Lab
Carleton University

Jason Jaskolka
jason.jaskolka@carleton.ca
@JasonJaskolka

CyberSEA Research Lab
https://carleton.ca/cybersea/
@CyberSEA_Lab