NANYANG TECHNOLOGICAL UNIVERSITY

School of Electrical & Electronic Engineering

EXQUISITUS, Centre for System Intelligence and Efficiency

Resilient Supervisory Control Against Smart Cyberattacks in Discrete-Event Dynamic Systems

Dr SU Rong (苏荣)

School of Electrical & Electronic Engineering

Nanyang Technological University, Singapore

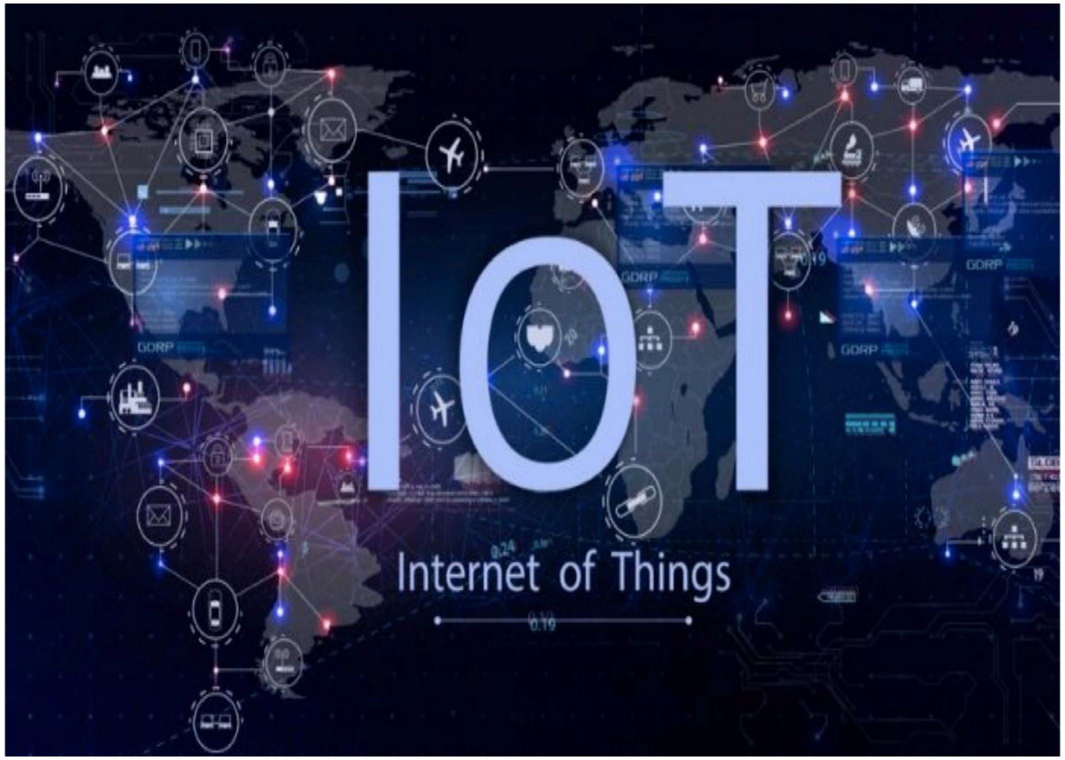
Tel.: +65 6790-6042, Email: rsu@ntu.edu.sg

Homepage: https://personal.ntu.edu.sg/rsu/

Outline

- ❑ **Introduction**
- ❑ Preliminaries on DES supervisory control
- ❑ Introduction to sensor attacks
- ❑ Introduction to actuator attacks
- ❑ An illustration example
- ❑ Conclusions

The Age of Networks



- **31B** IoT devices in 2020, **35B** in 2021, **75B** in 2025
- IoT adoptions in 2020[1]:
  - **93%** of enterprises;
  - **80%** of manufacturing companies
  - **90%** of cars connected to the web;
  - **3.5B** Cellular IoT connections installed.

A^4 = Anyone, Anything, Anywhere and *Anytime*

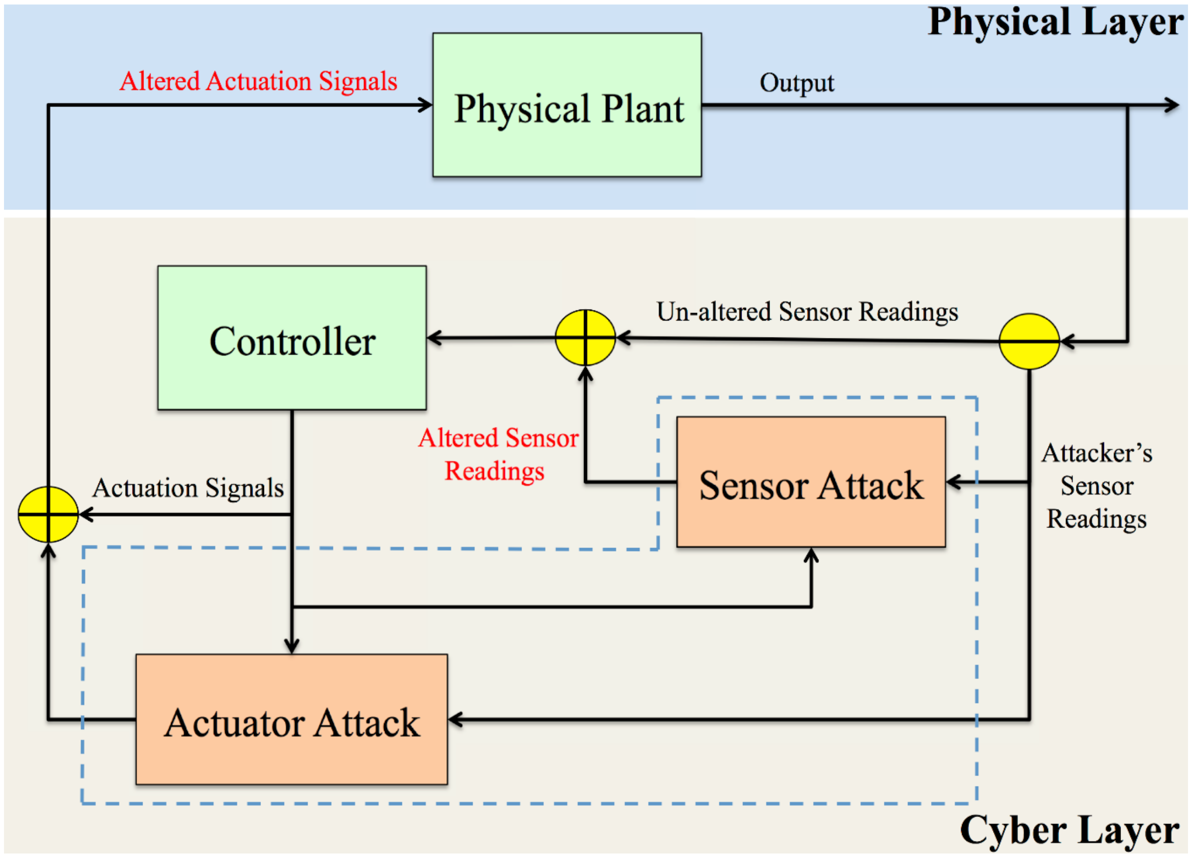
[1] Gilad David Maayan. The IoT rundown for 2020: stats, risks, and solutions, *Security Today*, 13/1/2020.

The Downside of Networked Society – Cybercrimes[2]

- Estimated cybercrime damages cost the world **$3 trillion** in 2015, and is expected to reach **$6 trillion** annually by 2021.
- **Yahoo hack** affected 3 billion users, and **Equifax breach** in 2017 affected 145.5 million customers. Others included WannaCry, NotPetya – **14 seconds** per ransom attack, cost **$5 billion** in 2017 in USA.
- Main types of attacks: **DDoS** attacks, **ransomware**, **zero-day exploits**.
- **Five most attacked industries** in 2015-2016 (and beyond)
  – Healthcare, manufacturing, financial, government, transportation.
  – Nearly 50% attacks were committed to small businesses.
  – Confidentiality, availability, authentication, integrity, non-repudiation.

[2] Steve Morgan. 2017 Official annual cybercrime report. *Cybercrime Magazine*. Oct. 16, 2017.

A Cyber-Physical System (CPS) Perspective



A generic CPS model[3]:

$$\dot{x} = f(t, x, u, w, \eta(t, \alpha, \beta))$$

- x: plant state
- u: control action
- w: disturbance
- η: cyber state function
- α: attacker's action
- β: defender's action

Possible control goals:

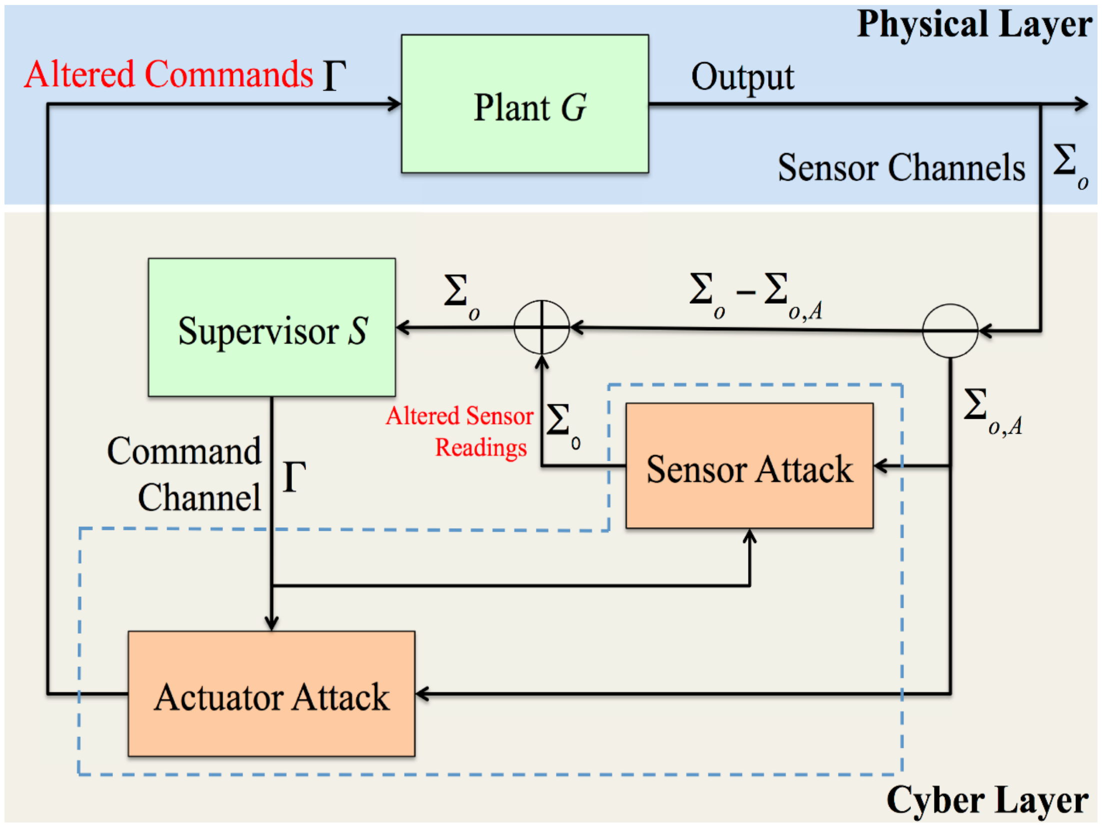
- To keep $x(t)$ in some D.
- To reach D optimally.

[3] Seyed Mehran Dibaji, Mohammad Pirani, David Bezalel Flamholz, Anuradha M. Annaswamy, Karl H. Johansson, Aranya Chakraborty . A systems and control perspective of CPS security. *Annual Reviews in Control*, vol. 47, pp. 394-411, 2019.

A Discrete-Event System (DES) View of CPS

- A DES is event driven, usually with a discrete set of states and events.
- A DES describes the **functional** evolution of a system.
- DES is common in industry, e.g.,
  - Manufacturing, logistics, medicare, robotics, transportation, etc.
- DES theory is part of some important research areas, e.g.,
  - Hybrid systems, multi-agent systems, robotics, formal method for controller synthesis, etc.
- A DES is vulnerable to cyber (sensor and actuator) attacks, which aim to **change the execution order of functions** to inflict damages.

A DES-based CPS Perspective



A DES-based CPS model:

$$
\begin{aligned}
x^+ &= f(x,u)\\
y &= g(x,u)\\
u &\in \beta(S(\alpha(y^-)),y^-)\\
y_0 &= \varepsilon \text{ (empty string)}
\end{aligned}
$$

- $x(x^+)$: current (next) state
- u: control action
- $y(y^-)$: current (past) output
- α: sensor attack function
- β: actuator attack function

Possible control goals:

- To keep x in some D.
- To reach D optimally.

Existing Cyber Security Research in DES

Existing research works:

- Fault tolerant control
- Opacity analysis and enforcement
- Discrete-event simulation of cyber attacks
- Game theoretical control for attack resilience in DES
- Supervisory control for attack resilience in DES

We are particularly interested in the following questions:

- What are characteristics of "**smart**" attacks?
- How to defend systems against "**smart**" attacks?

孙子 (Sun Tzu, 544 – 496 BC)

知己知彼，百战不殆!

– 孙子

If you know the enemy and know yourself, you need not fear the result of a hundred battles.

– Sun Tzu

Outline

- ❑ Introduction
- ❑ **Preliminaries on DES supervisory control**
- ❑ Introduction to sensor attacks
- ❑ Introduction to actuator attacks
- ❑ An illustration example
- ❑ Conclusions

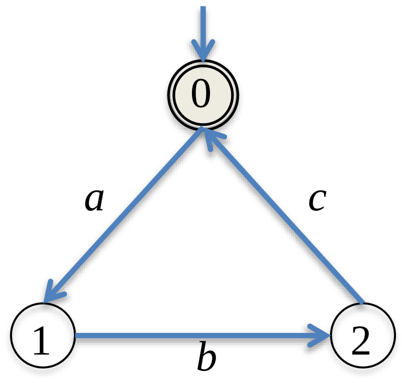
Languages and Projection

- Let Σ^* be the free monoid over a finite alphabet Σ, where
  - Each element in Σ^* is a *string*, and each subset $L \subseteq \Sigma^*$ is a *language*.
  - The unit element is ε, which is also called the *empty* string.
  - The monoid binary operation is *concatenation*, i.e., $(\forall s,t \in \Sigma^*) st \in \Sigma^*$.
  - We use $s \le s'$ to denote that s is a *prefix* of s', i.e., $(\exists t \in \Sigma^*) st = s'$. Write $s'/s = t$.
  - Prefix closure: $\overline{L} = \{s \in \Sigma^* \mid (\exists t \in \Sigma^*) st \in L\}$
  - Given two languages $U,V \subseteq \Sigma^*$, let $UV := \{st \in \Sigma^* \mid s \in U \wedge t \in V\}$.
- Let $\Sigma' \subseteq \Sigma$. The map $P : \Sigma^* \to \Sigma'^*$ is the *natural projection* w.r.t. (Σ, Σ'), if
  - $P(\varepsilon) = \varepsilon$;
  - $(\forall \sigma \in \Sigma) P(\sigma) = \begin{cases} \sigma & if\, \sigma \in \Sigma \setminus \Sigma' \\ \varepsilon & if\, \sigma \in \Sigma' \end{cases}$,
  - $P(s\sigma) = P(s)P(\sigma)$.

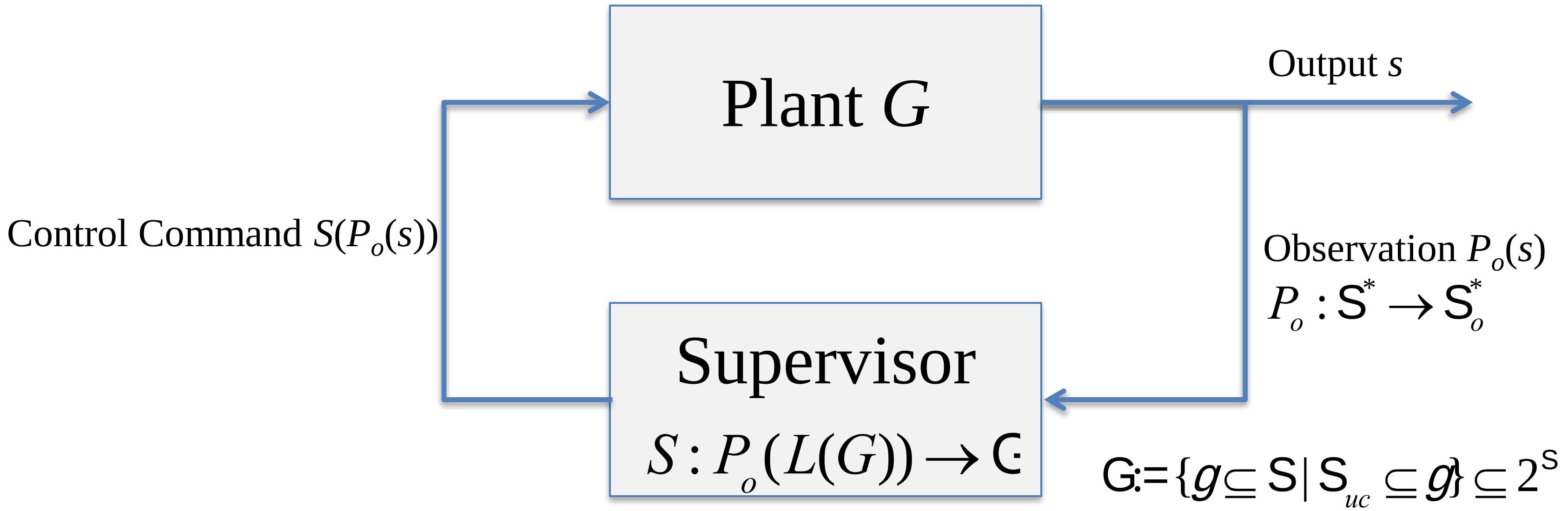
Finite-State Automaton

A *finite-state automaton* is a 5-tuple $G = (X, \Sigma, \xi, x_0, X_m)$, where

- X - the state set,
- X_m - the marker (or final) state set,
- Σ - the alphabet,
- $\xi : X \times \Sigma \to X$ - the (partial) transition map,
- x_0 - the initial state.
- The *closed* behavior: $L(G) = \{s \in \Sigma^* \mid \xi(x_0, s) \text{ is defined}\}$ [all tasks]
- The *marked* behavior: $L_m(G) = \{s \in L(G) \mid \xi(x_0, s) \in X_m\}$ [all complete tasks]



A Closed-Loop Discrete-Event System

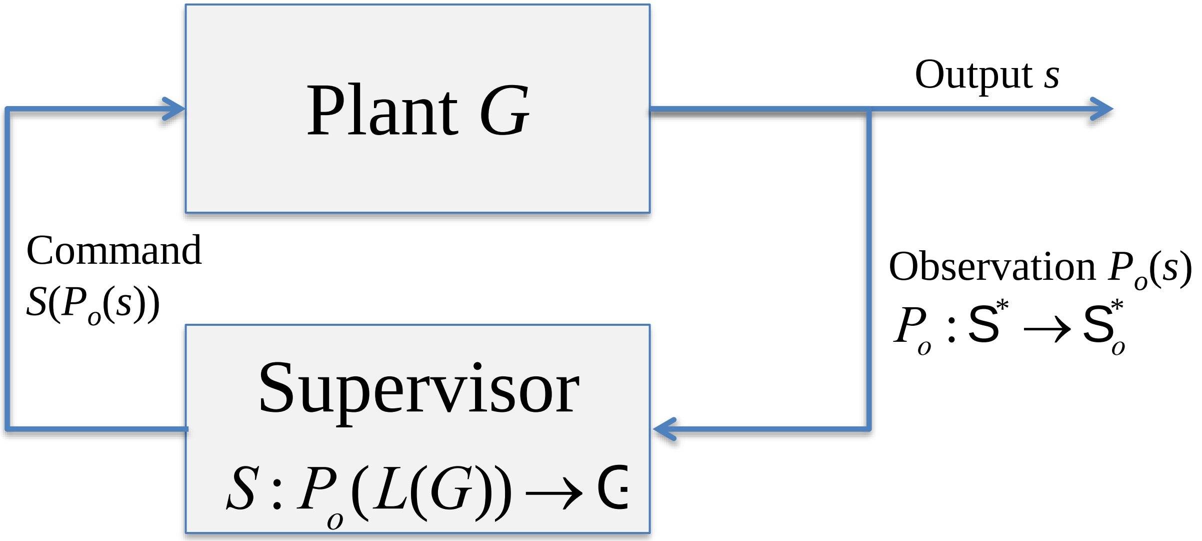

Plant G
Output s
Control Command $S(P_o(s))$
Observation $P_o(s)$
$P_o : S^* \rightarrow S_o^*$
Supervisor
$S : P_o(L(G)) \rightarrow G$
$G := \{g \subseteq S \mid S_{uc} \subseteq g\} \subseteq 2^S$

- Event partitions: $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc} = \Sigma_o \dot{\cup} \Sigma_{uo}$
- Control command (or pattern): $(" \ s \in L(G))S_{uc} \subseteq S(P_o(s))$
- Behaviors of closed-loop system *S*/*G* of the plant *G* under the control of *S*:
  - $e\hat{I} \ \ L(S/G)$
  - $(" \ s \in L(V/G))(" \ s \in S)ss \in L(V/G) \Leftrightarrow ss \in L(G) \wedge s \in S(P_o(s))$
  - $L_m(S/G) = L(S/G) Ç L_m(G)$

Ramadge-Wonham Supervisory Control Problem[4]

P. J. Ramadge

W. M. Wonham



Given a plant G and a requirement $E \subseteq L_m(G)$, find a supervisor S such that

- $L_m(S/G) \subseteq E$ [The closed-loop system satisfies the requirement E.]
- $L(S/G) = \overline{L_m(S/G)}$ [Each incomplete task in S/G can be completed.]
- $(" s')L_m(S'/G) \subseteq L_m(S/G)$ [The closed-loop system should be least restrictive.]

[4] P. J. Ramadge, W. M. Wonham. Supervisory control of a class of discrete-event systems. *SIAM Journal on Control and Optimization*, vol. 25, no. 1, pp. 206-236, 1987.

Outline

❑ Introduction

❑ Preliminaries on supervisory control

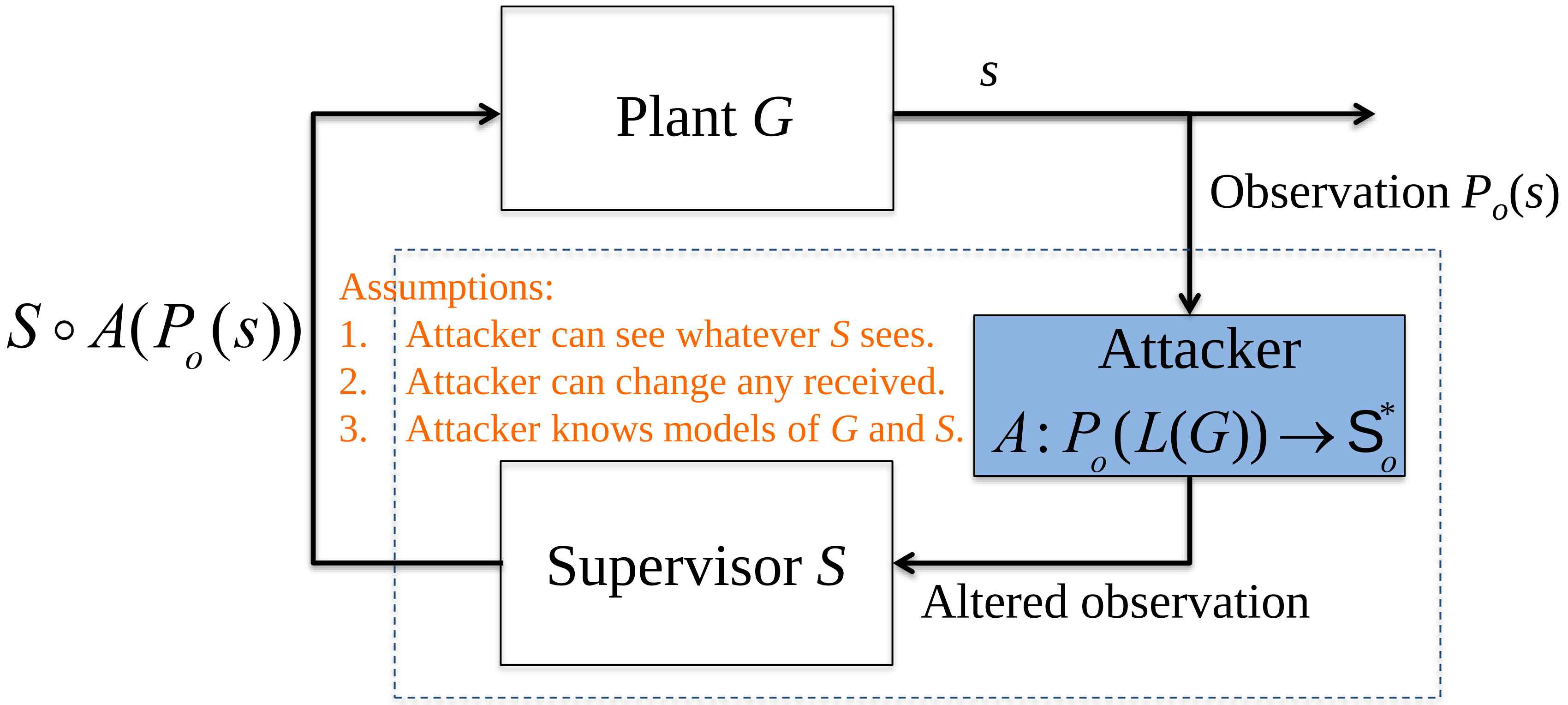
❑ **Introduction to sensor attacks**

❑ Introduction to actuator attacks

❑ An illustration example

❑ Conclusions

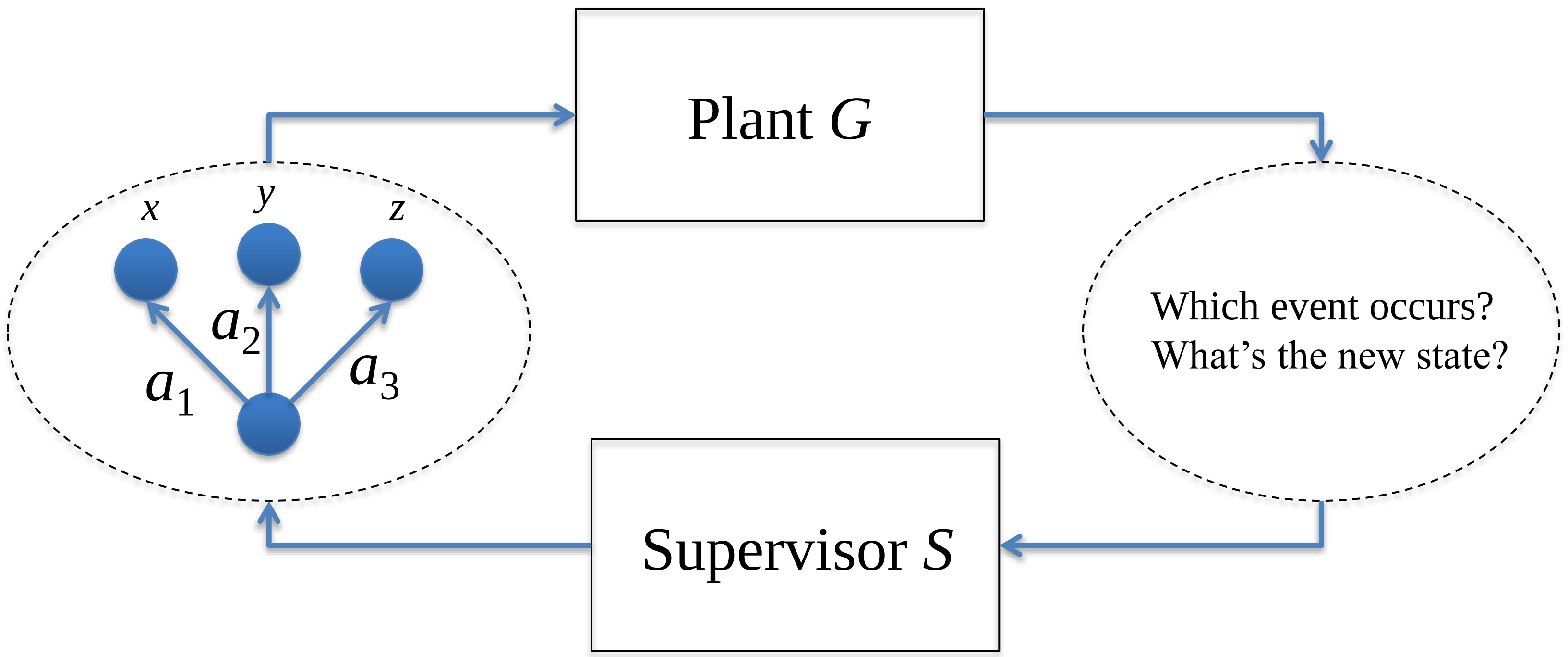
A Simple Architecture of Sensor Attack



Assumptions:
1. Attacker can see whatever S sees.
2. Attacker can change any received.
3. Attacker knows models of G and S.

- The composition $S \circ A$ is essentially a new supervisor.
- Thus, the new closed-loop system $S \circ A / G$ is defined as usual.
- **Question: What requirements does A need to satisfy?**

Why Attack on Supervisor is Possible?



- Non-determinism involved in event firing
- Observation based state estimation

} **Vulnerability**

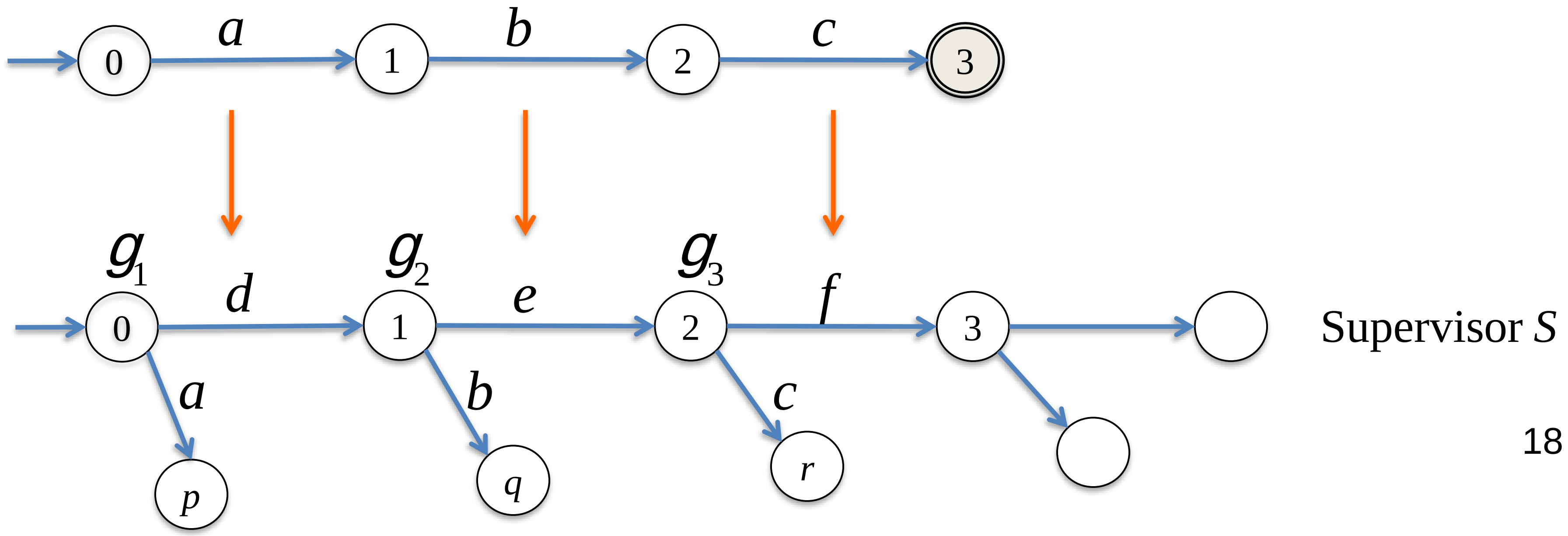
NANYANG TECHNOLOGICAL UNIVERSITY

School of Electrical & Electronic Engineering

Intuitive Illustration

Assume that an attacker *A* wants to achieve a string *abc*.

- Assume that *a* Î g_1, *b* Î g_2, *c* Î g_3.
- The attacker replaces *a* with *d* to trick the supervisor *S* to issue γ_2.
- Then the attacker replaces *b* with *e* to trick *S* to issue γ_3.
- The attacker could continue this trick as long as it is possible.



An Attack Model[5]

An attack model for G is a map $A: P_o(L(G)) \rightarrow \Sigma_o^*$, where

- $A(\epsilon) = \epsilon$
- $(\forall s\sigma \in P_o(L(G)))A(s) \le A(s\sigma) \wedge |A(s\sigma)| - |A(s)| \le n$ for some $n \in N$

Let $\equiv_{N,G}$ denote the Nerode equivalence relation over P_o(L(G)), i.e.,

$$(\forall s, s' \in P_o(L(G)))s \equiv_{N,G} s' \Leftrightarrow [(\forall t \in \Sigma_o^*)st \in P_o(L(G)) \Leftrightarrow s't \in P_o(L(G))]$$

The attack model A is *regular* with respect to $\equiv_{N,G}$, if

$$(\forall s\sigma, s'\sigma \in P_o(L(G)))s \equiv_{N,G} s' \Rightarrow A(s\sigma)/A(s) = A(s'\sigma)/A(s')$$

[5] R. Su. Supervisor synthesis to thwart cyber attack with bounded sensor reading alterations. *Automatica*, vol. 94, pp. 35-44, 2018.

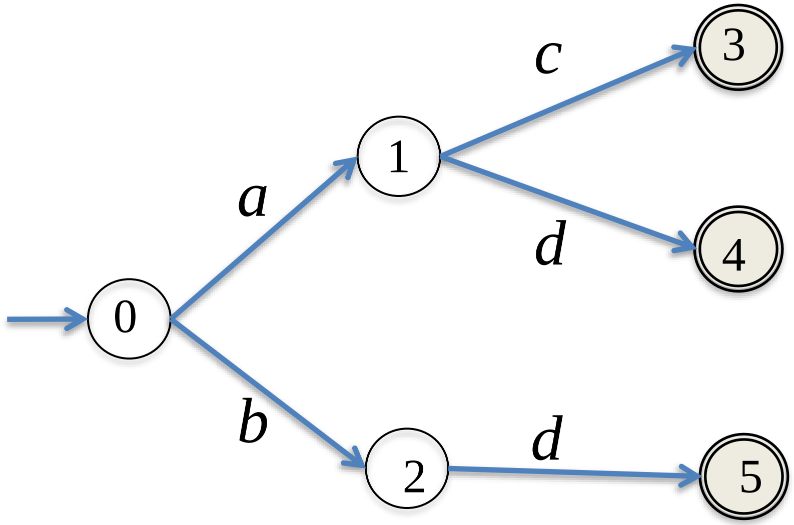
Closed-Loop System $S \circ A / G$

Since $S \circ A : P_o(L(G)) \to \Gamma : t \mapsto S \circ A(t) := S(A(t))$, we have

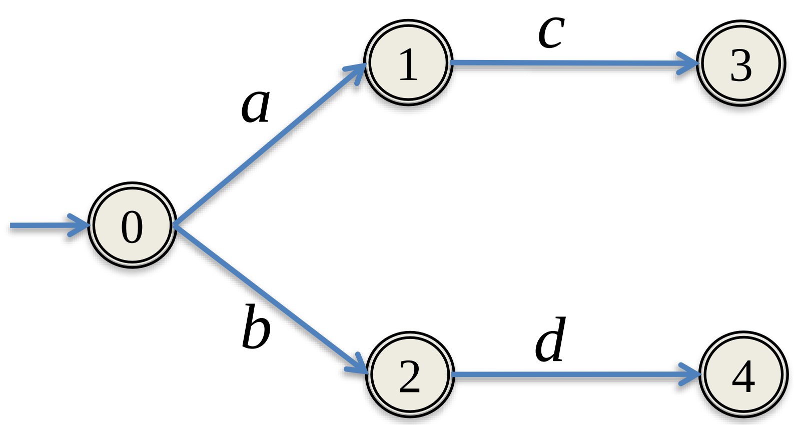
- $\varepsilon \in L(S \circ A / G)$
- $(\forall s \in L(S \circ A / G))(\forall \sigma \in \Sigma) s\sigma \in L(S \circ A / G) \Leftrightarrow s\sigma \in L(G) \wedge \sigma \in S \circ A(P_o(s))$
- $L_m(S \circ A / G) = L(S \circ A / G) \cap L_m(G)$

Assumption 3: Both *A* and *S* are regular with respect to $\circ_{N,G}$.

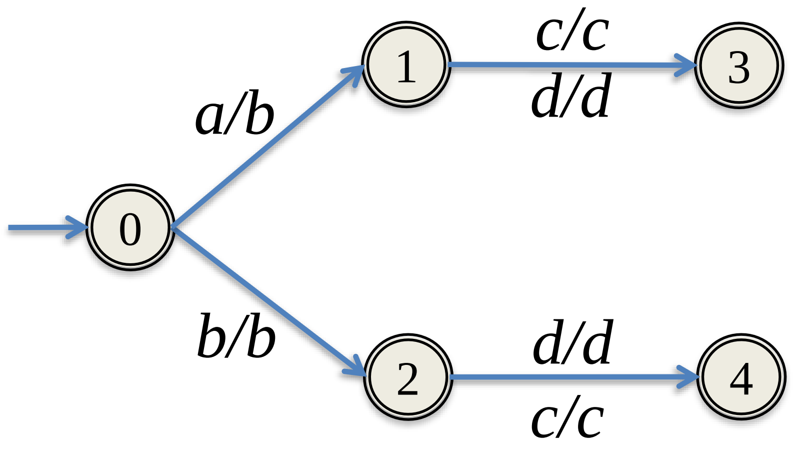
Example 1



Plant G

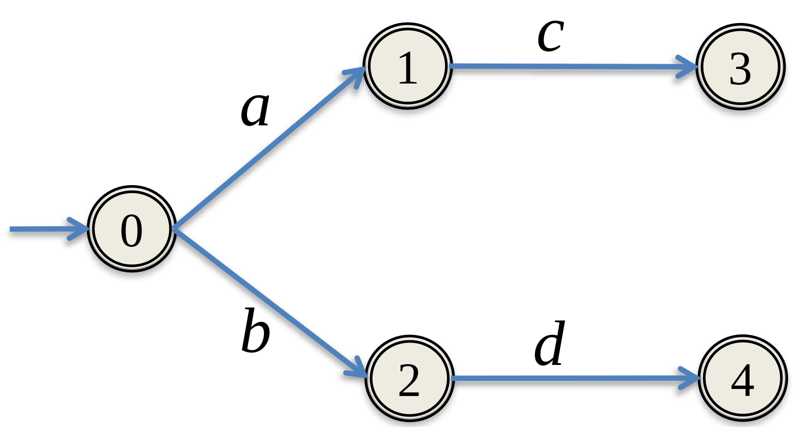


Supervisor S

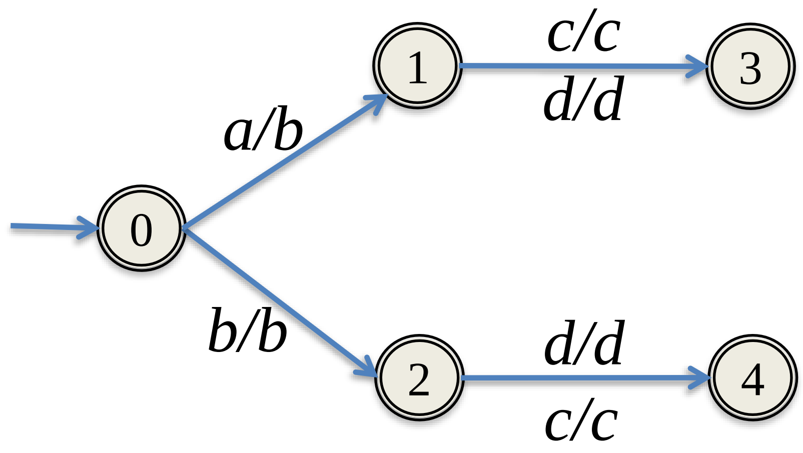


Attacker A

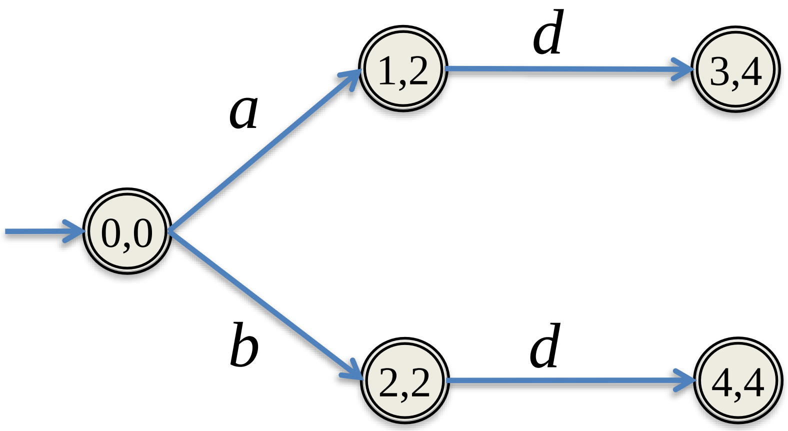
Example 1 – Sequential Composition



Supervisor S



Attacker A



$S \circ A$

Example 1 – Closed-Loop Behavior

Plant G

$$\begin{tikzcd}
 & & & 3 \\
 & & 1 \arrow[ur, "c"] \arrow[dr, "d"'] & \\
 {} \arrow[r] & 0 \arrow[ur, "a"] \arrow[dr, "b"'] & & 4 \\
 & & 2 \arrow[r, "d"] & 5
\end{tikzcd}$$

$S \circ A$

$$\begin{tikzcd}
 & & 1,2 \arrow[r, "d"] & 3,4 \\
 {} \arrow[r] & 0,0 \arrow[ur, "a"] \arrow[dr, "b"'] & & \\
 & & 2,2 \arrow[r, "d"] & 4,4
\end{tikzcd}$$

$$\begin{tikzcd}
 & & 1,2,1 \arrow[r, "d"] & 3,4,4 \\
 {} \arrow[r] & 0,0,0 \arrow[ur, "a"] \arrow[dr, "b"'] & & \\
 & & 2,2,2 \arrow[r, "d"] & 4,4,5
\end{tikzcd}$$

$S \circ A / G$

Smart Sensor Attack

Definition 1

A closed-loop system (G,S) is *attackable* if there exists a non-empty attack model A such that the following properties hold:

(1) Covertness: $$A(P_o(L(G))) \subseteq P_o(L(S/G)) \qquad (1)$$

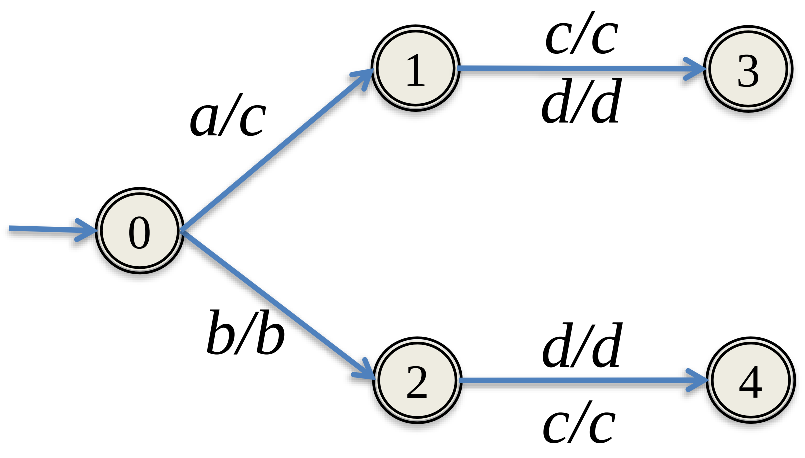
(2) Damage infliction: Let $\underline{L_{dam} := L(G) - L(S/G)}$.

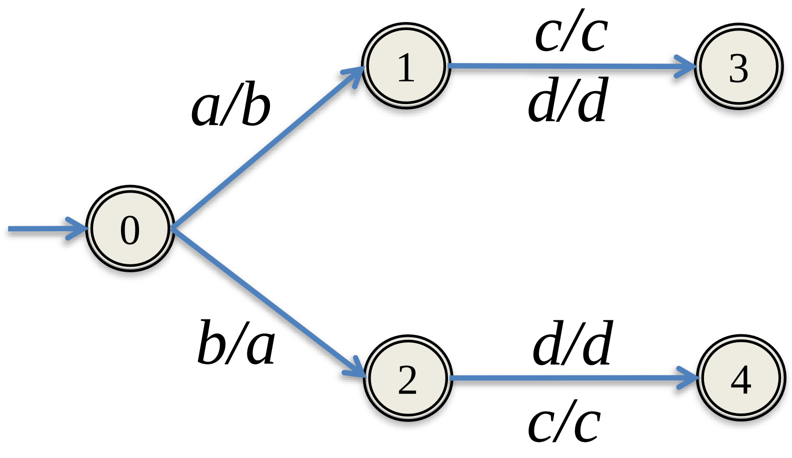
[Strong] $$L(S \circ A/G) = L(S \circ A/G) \cap L_{dam} \qquad (2\text{-}1)$$

[Weak] $$L(S \circ A/G) \cap L_{dam} \neq \varnothing \qquad (2\text{-}2)$$

$L(S \circ A/G)$ satisfying (1)-(2) is a *smart sensor attack language*.

Example 1 - Revisit





Attack Model A_1

Attack Model A_2

Not covert!
Not inflict any damage!
Thus, it is not smart!

A smart sensor attack

Supremal Smart Sensor Attack Language

Given a set of all smart sensor attacks $\{A_i \mid i \in I\}$ of (*G*, *S*), let

$$\bigvee_{i\in I} A_i : P_o(L(G)) \to 2^{\Sigma_o^*} : t \mapsto \bigvee_{i\in I} A_i(t) := \{A_i(t) \mid i \in I \wedge t \in L(S \circ A_i / G)\},$$

and we have

$$\bigvee_{i\in I} A_i(P_o(L(G))) = \bigcup_{i\in I} A_i(P_o(L(G))).$$

Let

$$S \circ (\bigvee_{i\in I} A_i) : P_o(L(G)) \to 2^{\Gamma} : t \mapsto S \circ (\bigvee_{i\in I} A_i)(t) := \{S \circ A_i(t) \mid i \in I \wedge t \in L(S \circ A_i / G)\},$$

and we can derive that

$$L(S \circ (\bigvee_{i\in I} A_i) / G) = \bigcup_{i\in I} L(S \circ A_i / G).$$

All three conditions in Def. 1 holds for $A := \bigvee_{i\in I} A_i$. Clearly, we have

$$(\forall i \in I) L(S \circ A_i / G) \subseteq L(S \circ A / G).$$

$L(S \circ A / G)$ is called the *supremal* smart sensor attack language.

Supremal Smart Sensor Attack Language (cont.)

Theorem 1

Given a closed-loop system (*G*, *S*) and a protected observation alphabet $\Sigma_{o,p}$, the existence of a regular smart strong sensor attack model is decidable. In case the supremal regular smart attack language exists, it is computable with the following complexity:

$$O(2^{3|G||S|^2}|\mathrm{S}||\mathrm{D}_n|) = O(2^{3|G||S|^2}|\mathrm{S}||\mathrm{S}_o|^n),$$

where Δ_n is the set of all observable strings whose lengths are no more than *n*.

Resilience against Smart Sensor Attacks

Problem 1: [RSaRSSA]

Given a plant *G* and a requirement *E*, decide whether there exists a regular and normal supervisor *S* to avoid any regular smart sensor attack *A* that inflicts a **weak** damage, i.e.,

$$L(S \circ A / G) \cap (L(G) - L(S / G)) \neq \varnothing.$$

[strong damage Þ weak damage, no weak damage Þ security]

Problem 2:

If the answer to Problem 1 is *yes*, compute one such supervisor *S*.

Theorem 2

Given a closed-loop system (*G*, *S*), the existence of a regular smart sensor attack *A* for weak damage with respect to $\Sigma_{o,p}$ is decidable.

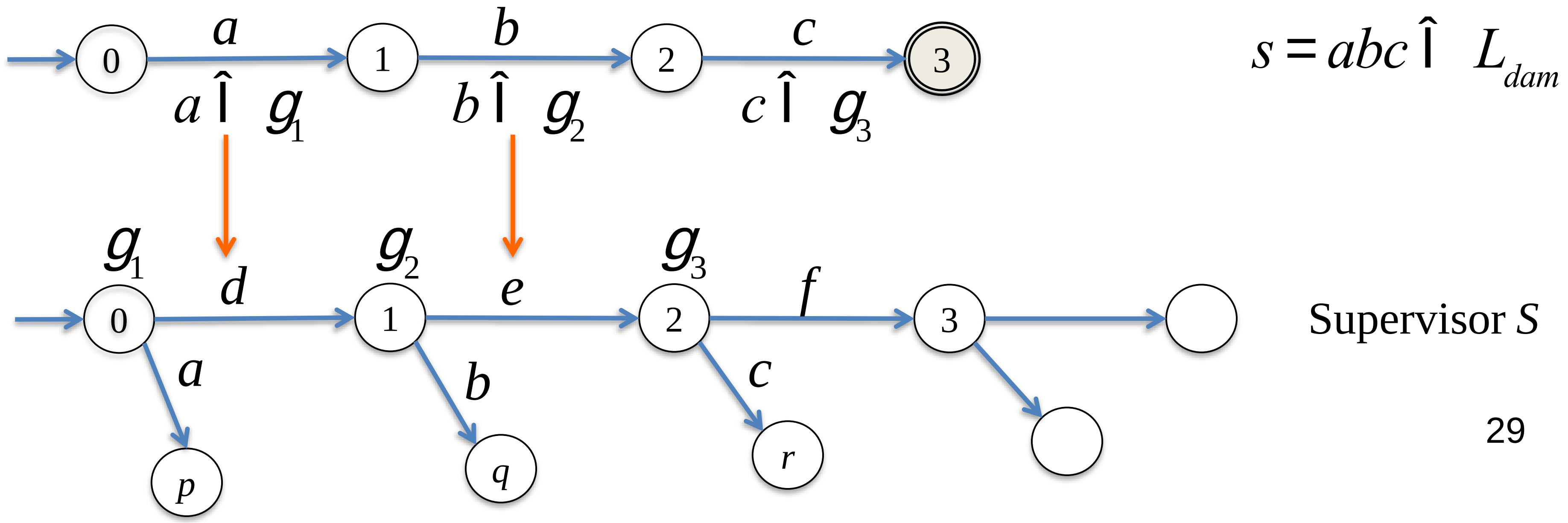
$abc \in L_{dam}$

$$\underline{[(\varepsilon,\gamma_1)(a,\gamma_2)(b,\gamma_3)}\,;\underline{(\varepsilon,\gamma_1)(d,\gamma_2)(e,\gamma_3)]}$$

Risk Pair

What *A* needs.

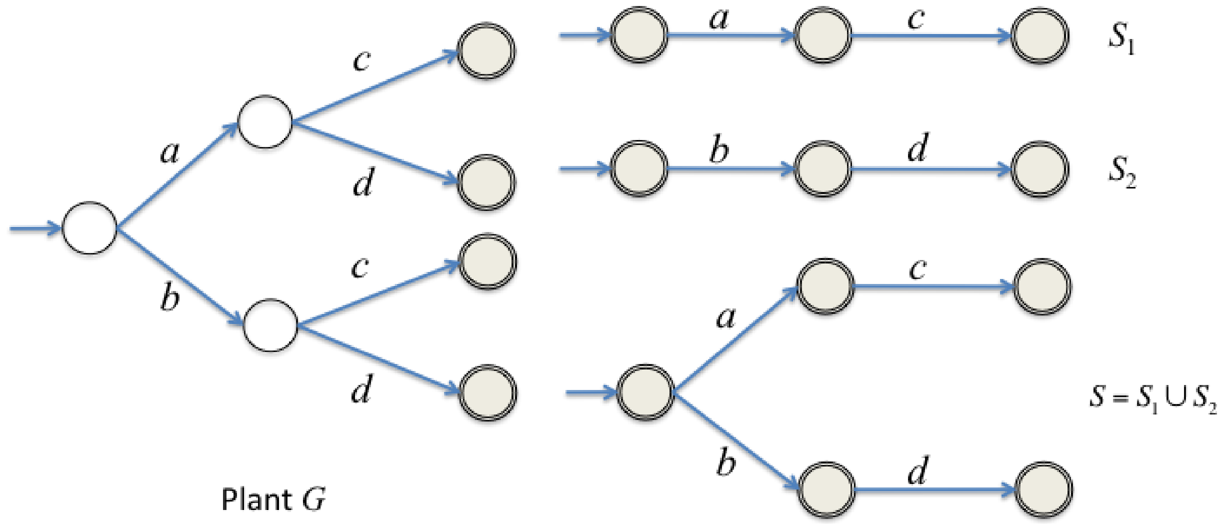
What *S* can supply.



Decidability of Existence of RSaRSSA

Theorem 3[6]

Given a plant G and a requirement E, let L_{dam} be a regular damage language. Then the existence of a solution of RSaRSSA in Problem 1 is decidable. In the case that there is a solution to Problem 1, there is an algorithm to compute a maximally permissive RSaRSSA. But the least restrictive solution (or the supremal RSaRSSA) usually does not exist.



[6] R. Su, M. Reniers. On decidability of existence of nonblocking supervisors resilient to smart sensor attacks. *Automatica*, under review, 2021.

Outline

❑ Introduction

❑ Preliminaries on supervisory control

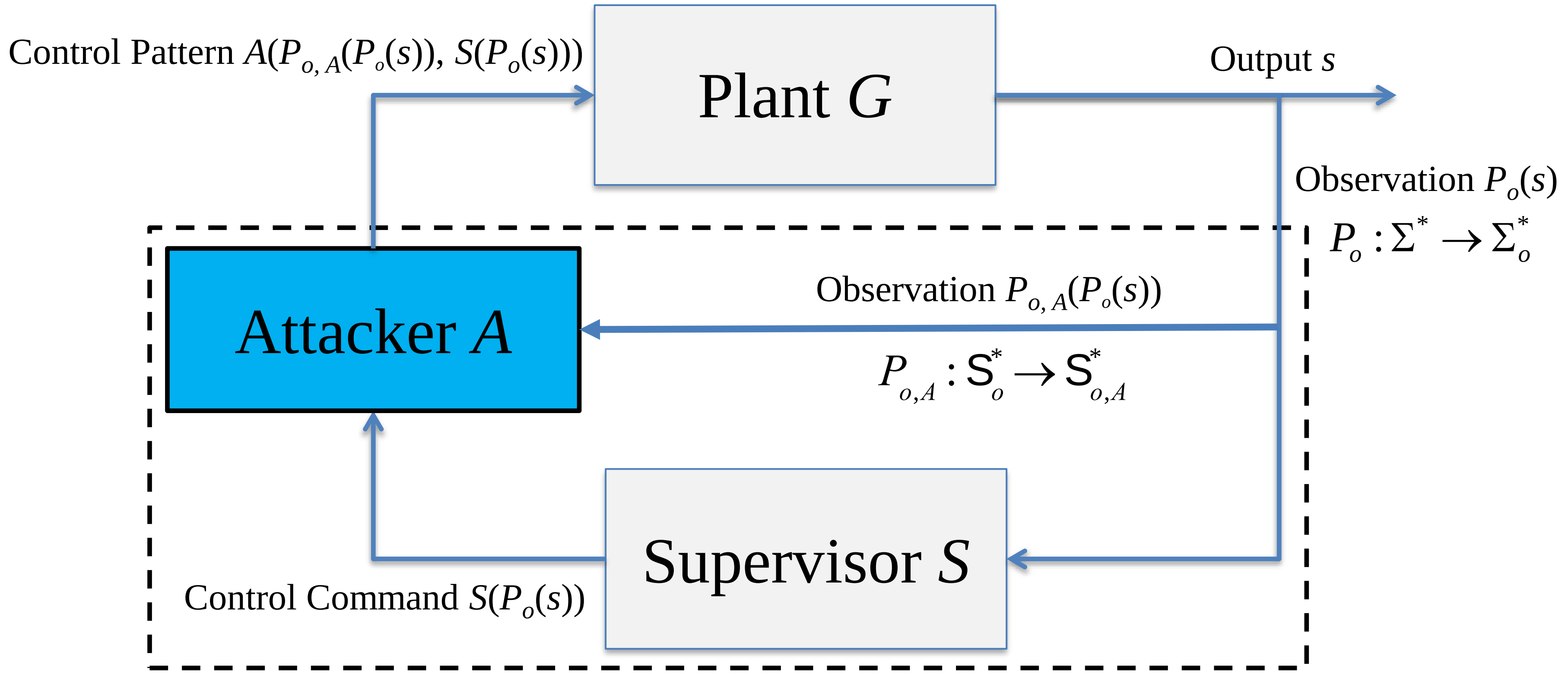
❑ Introduction to sensor attacks

❑ **Introduction to actuator attacks**

❑ An illustration example

❑ Conclusions

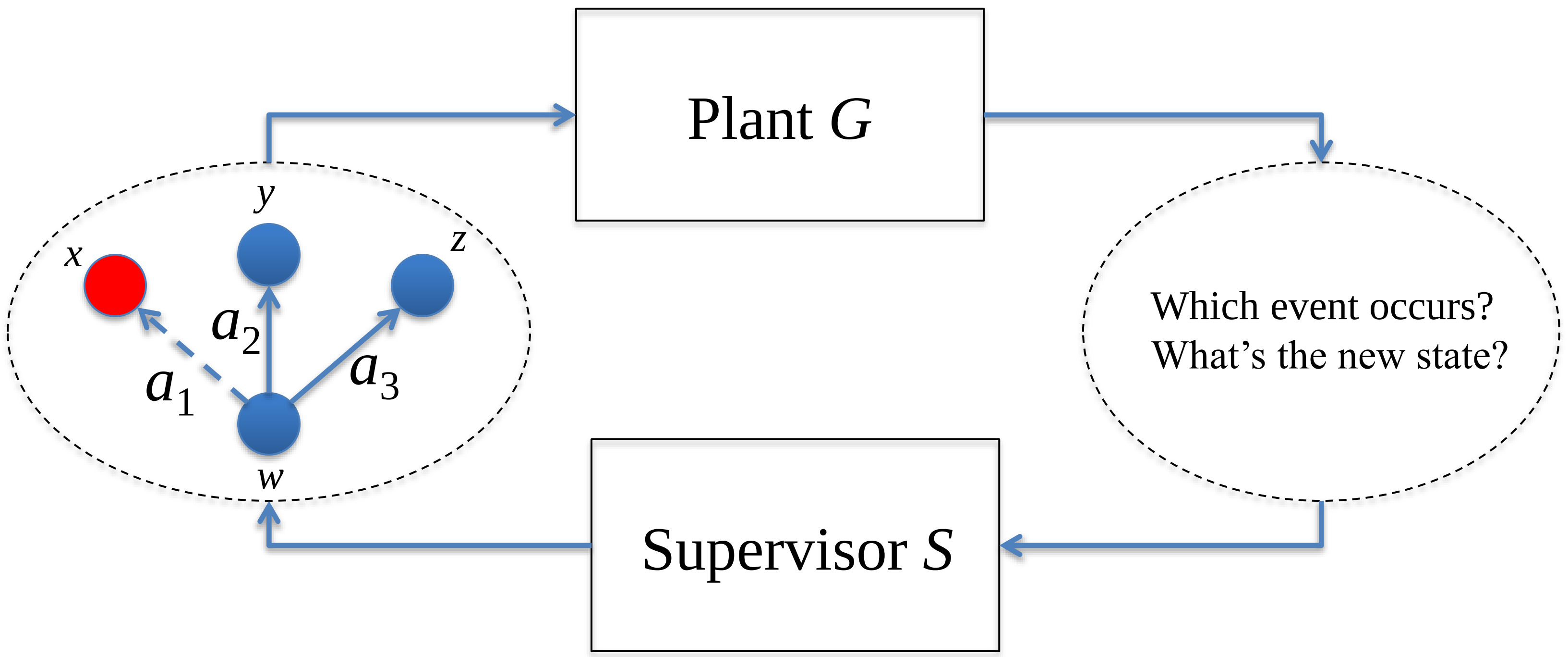
A Simple Architecture of Actuator Attack


Control Pattern $A(P_{o,A}(P_o(s)), S(P_o(s)))$
Plant G
Output s
Observation $P_o(s)$
$P_o : \Sigma^* \to \Sigma_o^*$
Attacker A
Observation $P_{o,A}(P_o(s))$
$P_{o,A} : \Sigma_o^* \to \Sigma_{o,A}^*$
Supervisor S
Control Command $S(P_o(s))$

Questions:

- What is the model A?
- What is the attacked supervisor $A \circ (P_{o,A}, S)$?
- What is the attacked closed-loop system $A \circ (P_{o,A}, S) / G$?

Why Actuator Attack on Supervisor is Possible?



- Existence of attackable actuation events
- Existence of "risky" states

} **Vulnerability**

Intuitive Illustration

Assume that an attacker wants the plant G to reach a damaging state 7, but can't observe events b and c. A supervisor can observe a, b, c.

- If the supervisor is S_1, then after observing a, it is safe for the attacker to initiate an actuator attack and enable event d.
- If the supervisor is S_2, then after observing a, the attacker can't initiate an actuator attack without having a risk of being detected.

$$\begin{tikzcd}
{} \arrow[r] & 1 \arrow[r, "c"] \arrow[dr, "b"'] & 2 \arrow[r, "a"] & 5 \arrow[r, "d"] & 7 \\
& & 3 \arrow[r, "a"] & 4 \arrow[r, "d"] & 6
\end{tikzcd}$$

Plant G

$$\begin{tikzcd}
{} \arrow[r] & 1 \arrow[r, "c"] \arrow[dr, "b"'] & 2 \arrow[r, "a"] & 5 \\
& & 3 \arrow[r, "a"] & 4 \arrow[r, "d"] & 6
\end{tikzcd}$$

Supervisor S_1

$$\begin{tikzcd}
{} \arrow[r] & 1 \arrow[r, "c"] \arrow[dr, "b"'] & 2 \arrow[r, "a"] & 5 \\
& & 3 \arrow[r, "a"] & 4
\end{tikzcd}$$

Supervisor S_2

Detected!

So S_1 admits an attack, but S_2 does not.

An Actuator Attack Model[7][8]

- Information for an attacker: $(\Sigma_{c,A}, \Sigma_{o,A})$, where $\Sigma_{c,A} \subseteq \Sigma_c \wedge \Sigma_{o,A} \subseteq \Sigma_o$.

- Attacker's observation map: Recall that Γ is the set of all control patterns (commands).

$$P^S_{o,A}:\ P_o(L(G)) \rightarrow ((\Sigma_{o,A} \cup \{\epsilon\}) \times \Gamma)^*$$

where for all $w = u_1\sigma_1 \ldots u_n\sigma_n u_{n+1} \in L(G) \cap (\Sigma^*_{uo}\Sigma_o)^*\Sigma^*_{uo}$,

$$P^S_{o,A}(w) := \begin{cases} (\varepsilon, S(\varepsilon)) & \text{if } w = \varepsilon, \\ (\varepsilon, S(\varepsilon))(P_{o,A}(\sigma_1), S(\sigma_1)) \cdots (P_{o,A}(\sigma_1 \cdots \sigma_n), S(\sigma_1 \cdots \sigma_n)) & \text{otherwise.} \end{cases}$$

- A sensor attack over the supervisor S is modeled by a function

$$A : P^S_{o,A}(P_o(L(G))) \rightarrow \Gamma$$

[7] L. Lin, S. Thuijsman, Y. Zhu, S. Ware, R. Su, M. Reniers. Synthesis of supremal successful actuator attackers on normal supervisors. *ACC'19*, pp. 5614-5619, 2019.

[8] L. Lin, Y. Zhu, R. Su. Synthesis of covert actuator attackers for free. *Journal of Discrete event dynamic systems: Theory and Applications*, vol. 30, pp. 561-577, 2020.

Smart Actuator Attack

- Attacked supervisor $A \circ S: \ P_o(L(G)) \longrightarrow \Gamma$, where

$$(\forall s \in P_o(L(G))) A \circ S(s) = A(P^S_{o,A}(s)).$$

- Attacked closed-loop behaviors: $L(A \circ S / G)$ and $L_m(A \circ S / G) := L(A \circ S / G) \cap L_m(G)$.

Definition 2:

A closed-loop system (G, S) is *attackable* if there exists a non-empty actuator attack model A such that the following properties hold:

1) **Controllability:** $(\forall s \in L(A \circ S / G))\{s\}(S(s) - \Sigma_{c,A}) \cap L(G) \subseteq L(A \circ S / G)$.

2) **Covertness:** $L(A \circ S / G) \subseteq L(S / G)(\{\varepsilon\} \cup \Sigma_{c,A}) \cap L(G)$.

3) **Damage-inflicting**: Let $L_{dmg} \subseteq (L(S / G)\Sigma_{c,A} - L(S / G)) \cap L(G)$,

- <span>Strong</span> condition: $L(A \circ S / G) = \overline{L(A \circ S / G) \cap L_{dam}}$
- Weak condition: $L(A \circ S / G) \cap L_{dmg} \neq \varnothing$.

Supremal Smart Actuator Attack Language

Given a set of all smart actuator attacks $\{A_i \mid i \text{ Î } I\}$ of (G, S), let

$$\vee_{i\in I} A_i : P^S_{o,A}(P_o(L(G))) \to \Gamma : t \mapsto \vee_{i\in I} A_i(t) := \{A_i(t) \mid i \in I \wedge t \in L(A_i \circ S / G)\},$$

and we have

$$\text{Ú}_{i \text{Î} I} A_i(P^S_{o,A}(P_o(L(G)))) = \text{È}_{i \text{Î} I} A_i(P^S_{o,A}(P_o(L(G)))).$$

Let

$$(\vee_{i\in I} A_i) \circ S : P_o(L(G)) \to 2^{2^\Sigma} : t \mapsto (\vee_{i\in I} A_i) \circ S(t) := \{A_i \circ S(t) \mid i \in I \wedge t \in L(A_i \circ S / G)\},$$

and we can derive that

$$L((\vee_{i\in I} A_i) \circ S / G) = \bigcup_{i\in I} L(A_i \circ S / G).$$

All three conditions in Def. 2 holds for $A := \text{Ú}_{i \text{Î} I} A_i$. Clearly, we have

$$(\forall i \in I) L(A_i \circ S / G) \subseteq L(A \circ S / G).$$

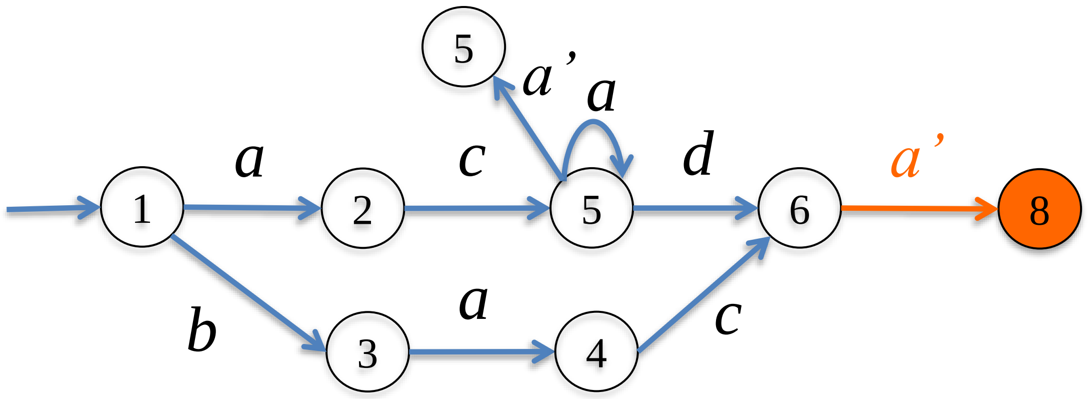
$L(A \circ S / G)$ is called the supremal *smart actuator attack language.*

Supremal Smart Actuator Attack Language (cont.)

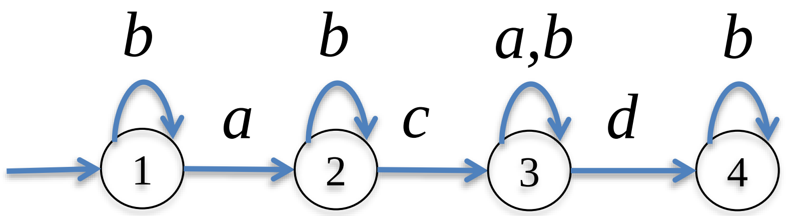
Theorem 4

Given a closed-loop system (*G*, *S*) and an attack tuple $(\Sigma_{c,A}, \Sigma_{o,A}, L_{dam})$, the supremal regular smart (strong or weak) actuator attack language exists and computable, whose complexity is exponential-time.

A Small Example

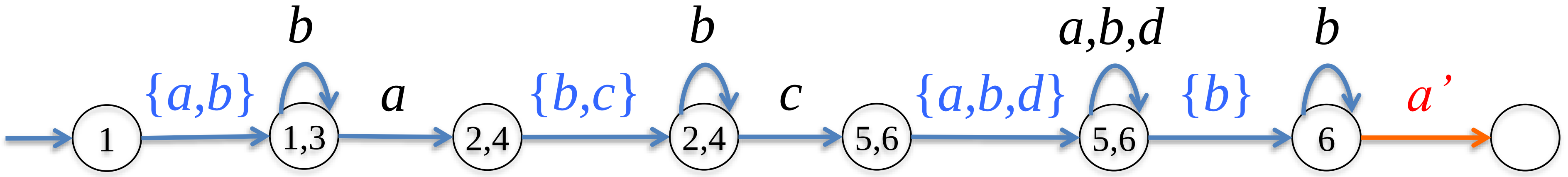


Plant G

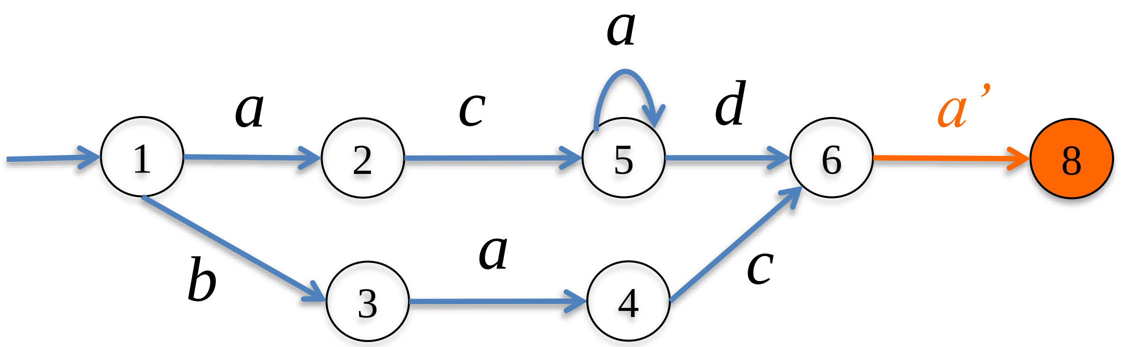


Supervisor S

$$L_{dam} = (aca^{*}d + bac)a', \mathrm{S} = \{a,b,c,d,a'\}, \mathrm{S}_o = \{a,c,d\}, \mathrm{S}_c = \mathrm{S}_{c,A} = \{a'\}, \mathrm{S}_{o,A} = \{a,c\}$$



Smart Actuator Attack A



$L(A \circ S / G)$

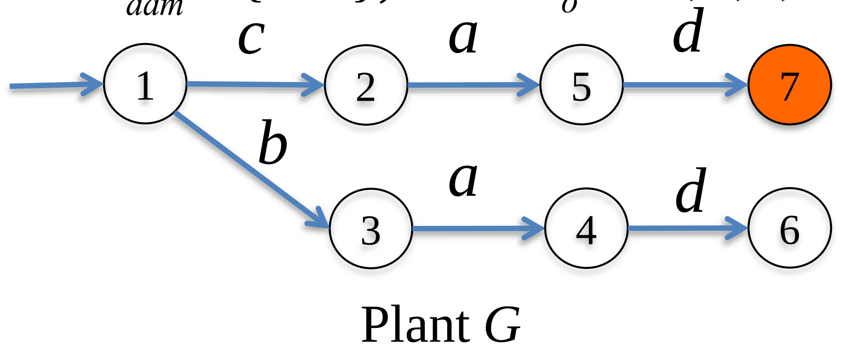
Supervisor Resilient to Smart Actuator Attack

CONJECTURE

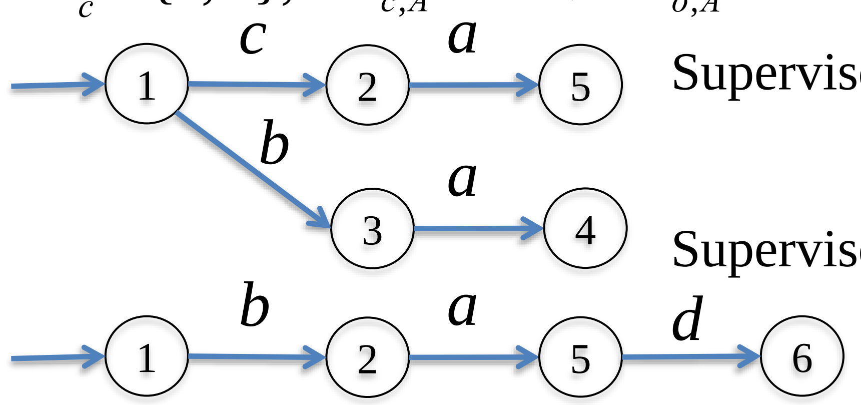
Given a plant G and a requirement E, let L_{dam} be a given regular damage language. Then the existence of a regular normal supervisor S, which does not admit any regular smart weak actuator attack w.r.t. $(\Sigma_{c,A}, \Sigma_{o,A}, L_{dam})$ is decidable.

The supremal one resilient to smart actuator attacks does not exist.

$L_{dam} = \{cad\}$, $\Sigma = \Sigma_o = \{a,b,c,d\}$, $\Sigma_c = \{c,d\}$, $\Sigma_{c,A} = \{d\}$, $\Sigma_{o,A} = \{a\}$



Plant G



Supervisor S_1

Supervisor S_2

Neither S_1 nor S_2 admits any smart actuator attack!
They both are maximal, but not comparable!

Resilient Supervisor Synthesis

Problem 3: [Resilient Supervisor Synthesis]

Given **plant** *G*, synthesize **supervisor** *S* over (Σ_c, Σ_o) such that there is no smart actuator **attack** over $(\Sigma_{c,A}, \Sigma_{o,A}, L_{dam})$.

Problem 4: [Supervisor Obfuscation]

Given **plant** *G* and **supervisor** *S*, synthesize **supervisor** *S'* over (Σ_c, Σ_o),

1) *S'* is control equivalent to *S*, i.e., $L(S/G) = L(S'/G)$.
2) There is no smart actuator **attack** over *S'*.

Some heuristic algorithms

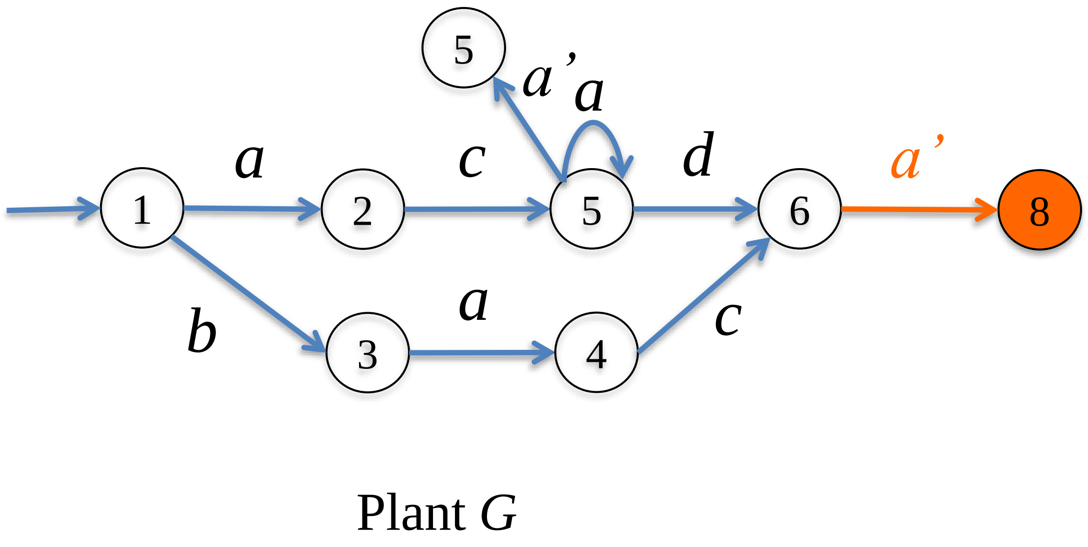
- SAT encoding of all *n*-bounded control-equivalent supervisors[9][10];
- Using sup-reduction to get minimum-state control-equivalent supervisors[11].

[9] L. Lin, Y. Zhu, R. Su. Towards bounded synthesis of resilient supervisors against actuator attacks. *IEEE CDC'19*, pp. 7659-7664, 2019.

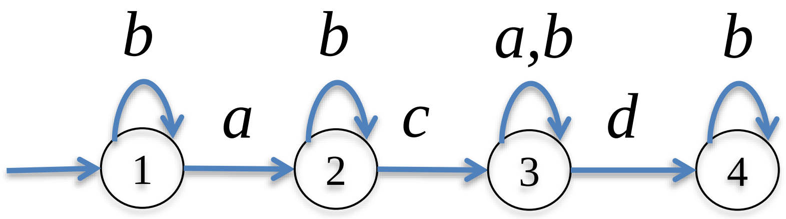
[10] L. Lin, R. Su. Bounded synthesis of resilient supervisors. *IEEE TAC*, accepted, 2021.

[11] Y. Zhu, L. Lin, R. Su. Supervisor obfuscation against actuator enablement attack. *ECC'19*, pp. 1760-1765, 2019.

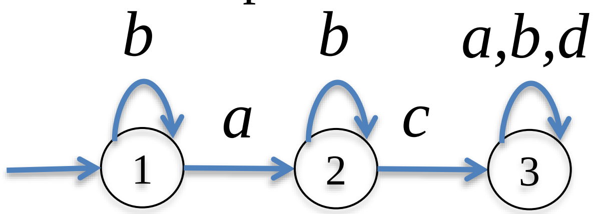
A Small Example -Revisit



Plant G

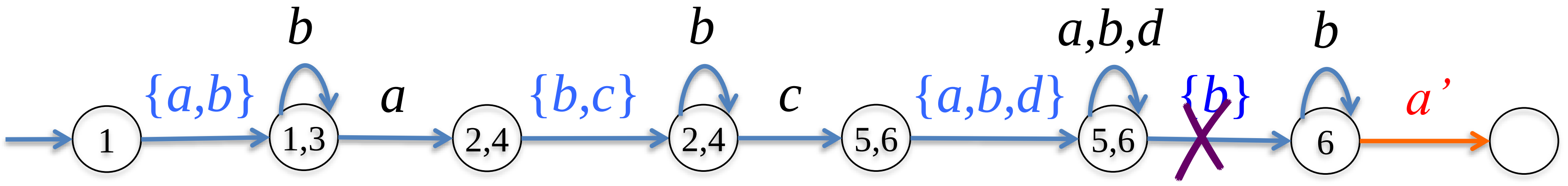


Supervisor S



Supervisor S'

$$L_{dam} = (aca^{*}d + bac)a', \mathsf{S} = \{a,b,c,d,a'\}, \mathsf{S}_o = \{a,c,d\}, \mathsf{S}_c = \mathsf{S}_{c,A} = \{a'\}, \mathsf{S}_{o,A} = \{a,c\}$$



There is NO smart actuator attack A on S'!

Outline

❑ Introduction

❑ Preliminaries on supervisory control

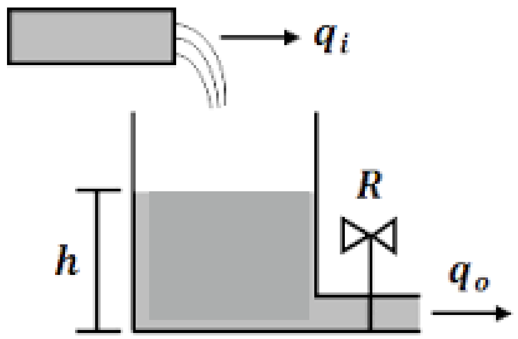
❑ Introduction to sensor attacks

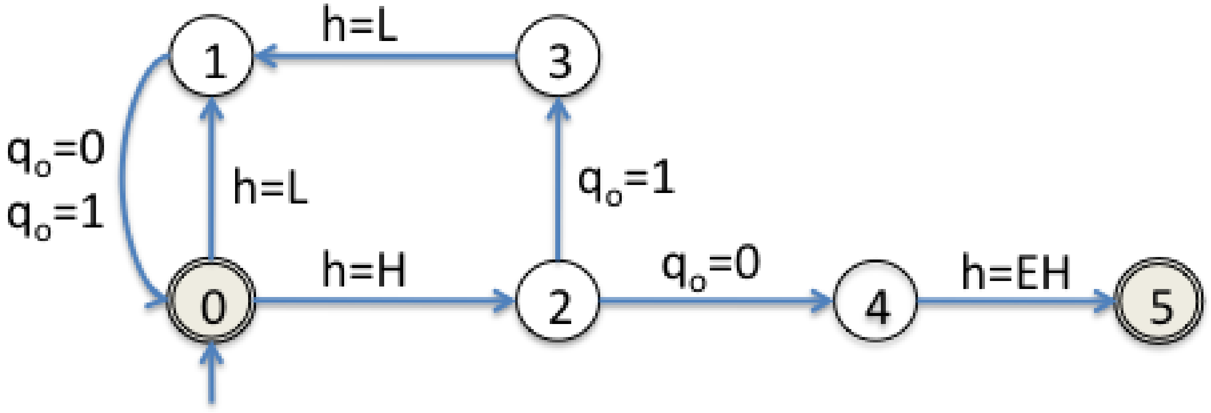
❑ Introduction to actuator attacks

❑ **An illustration example**

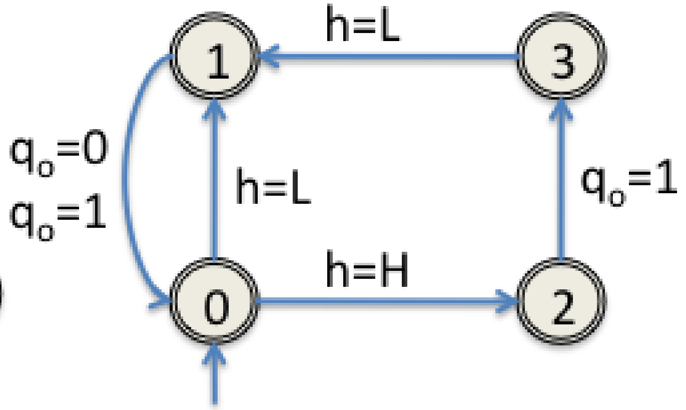
❑ Conclusions

A Small Tank Example





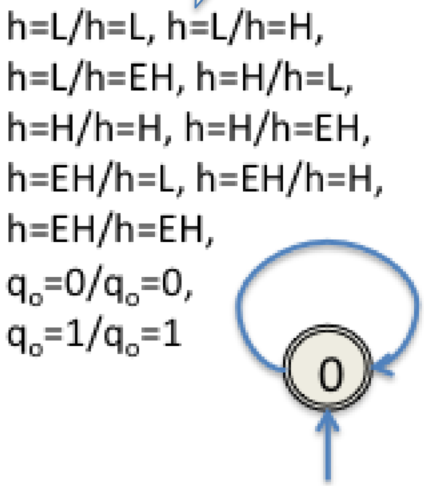
Plant G



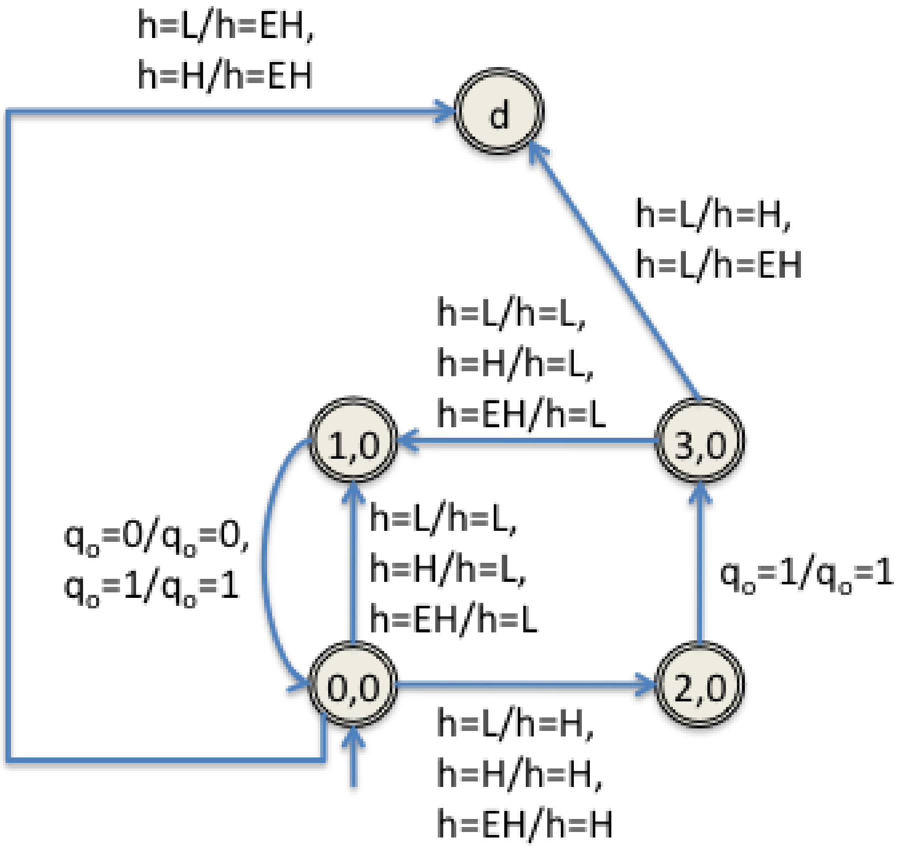
Supervisor S

Synthesis of A Regular Smart Sensor Attack Model – Step 1

Encode all possible sensor attack moves.

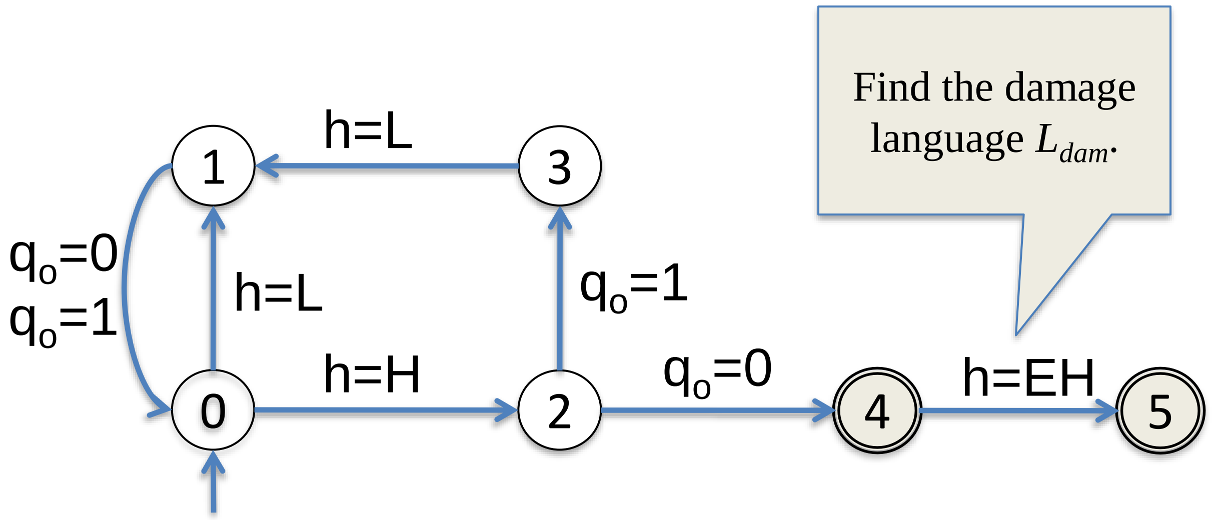


Attack Model A_0



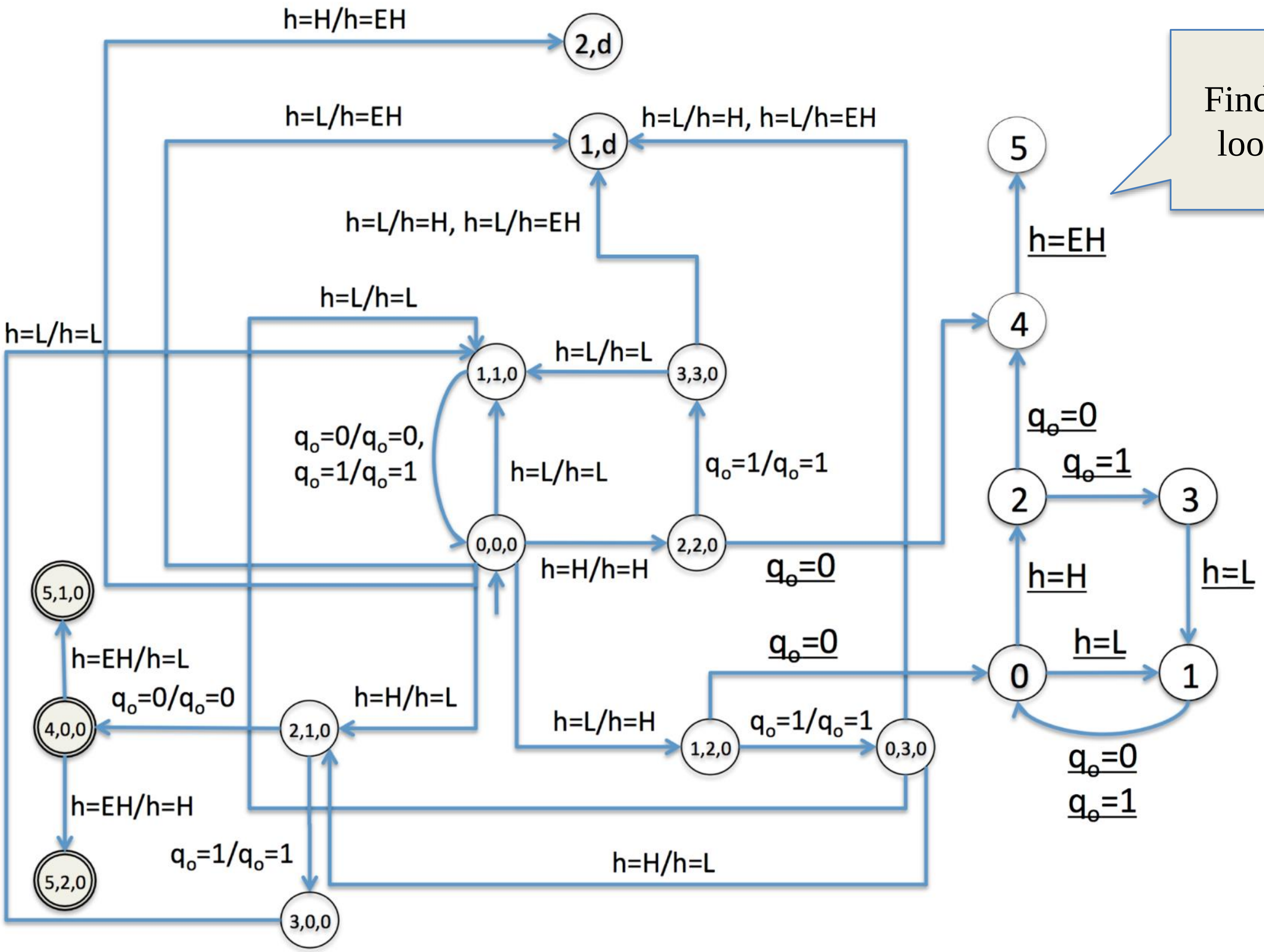
$A_0 \circ S$

Synthesis of A Regular Smart Sensor Attack Model – Step 2



Automaton E

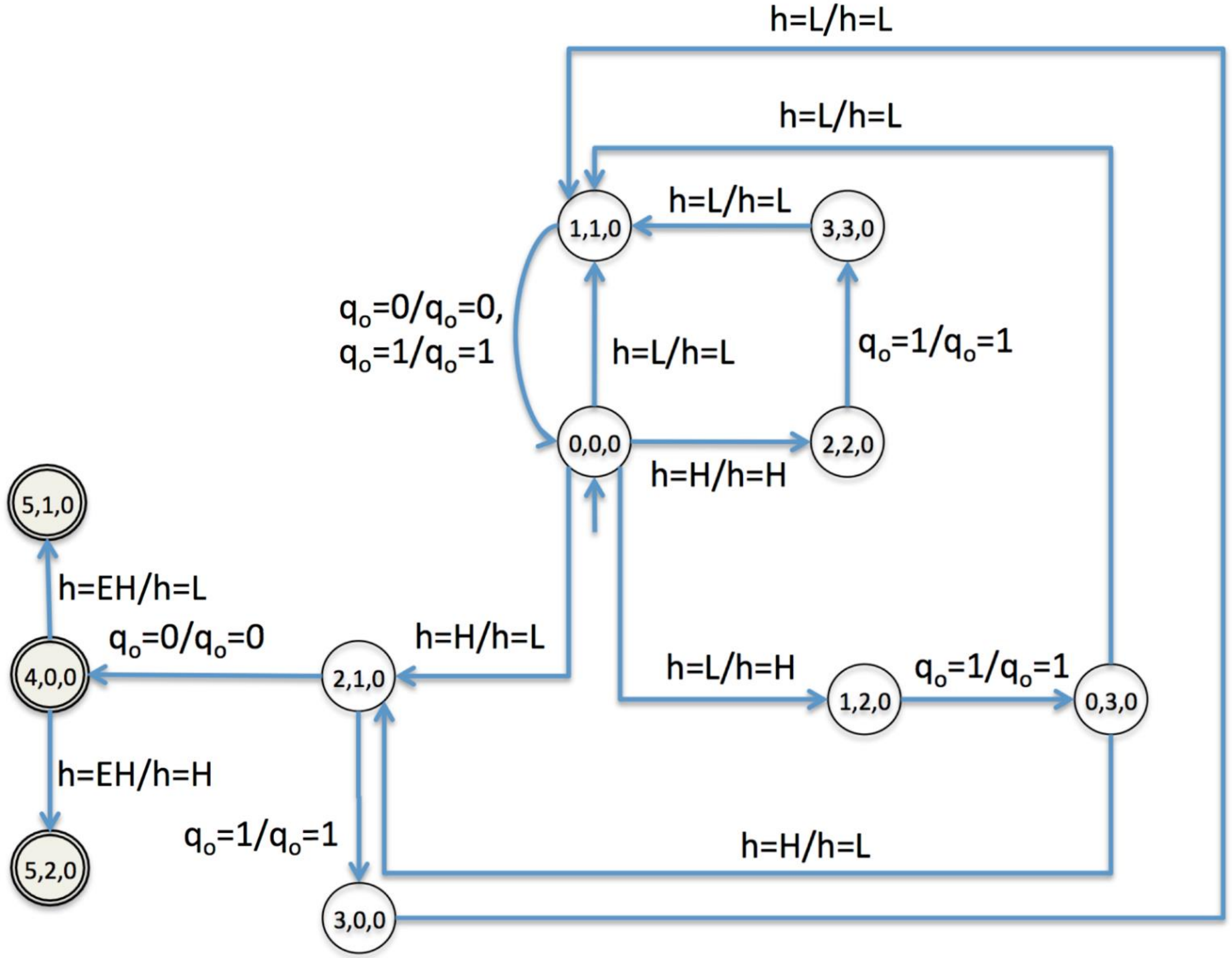
Synthesis of A Regular Smart Sensor Attack Model – Step 3



Find the closed-loop behavior.

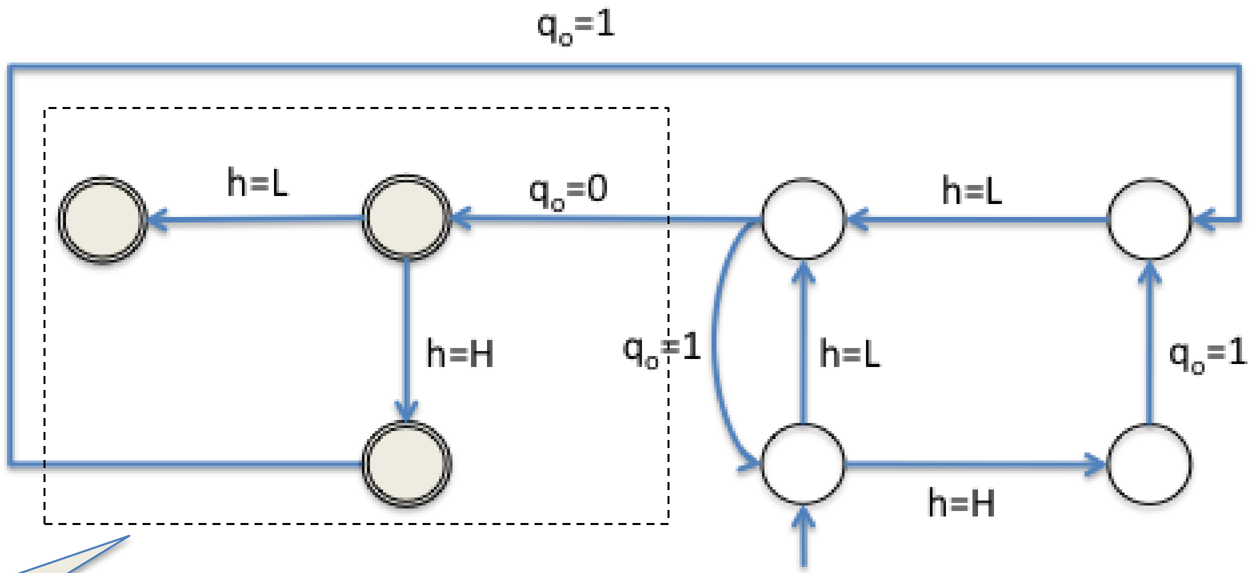
$$B := E \hat{\times} (A_0 \circ S)$$

Synthesis of A Regular Smart Sensor Attack Model – Step 4



Supremal Smart Sensor Attack Model *A*

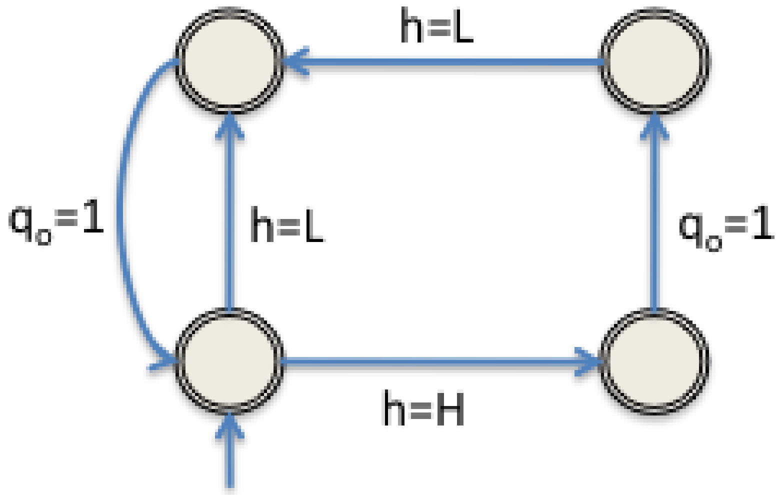
Synthesis of A Sensor-Attack Resilient Supervisor – Step 1



Identify risky strings in S that could be used by an attacker A.

Automaton Model of $q(L_m(A))$

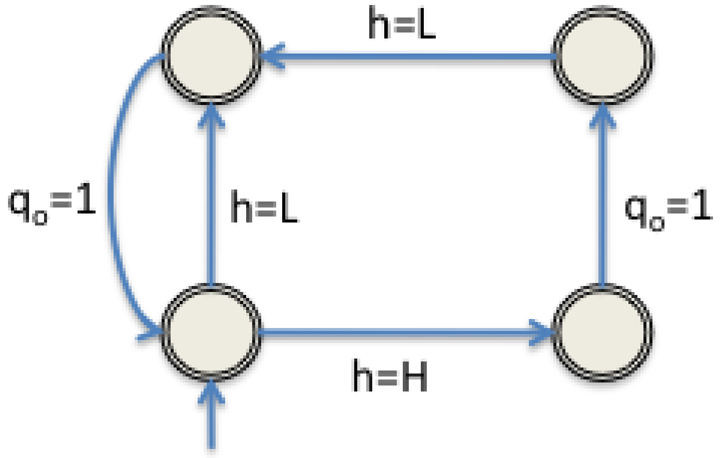
Synthesis of A Sensor-Attack Resilient Supervisor – Step 2



Remove all risky strings identified in Step 1.

Automaton Model of $L(\hat{S}) - \theta(L_m(A))\Sigma^*$

Synthesis of A Sensor-Attack Resilient Supervisor – Step 3



A Supervisor Resilient to Strong Smart Sensor Attacks

Simple Resilient Law: **DO NOT CLOSE DISCHARGE VALVE *R*!**

Conclusions

- Regular languages can be used to model sensor and actuator attacks.
- Supremal (sensor and actuator) attack languages exists.
- But supremal resilient supervisors typically do not exist.
- The current research has two major application potentials:
  - To determine risky system behaviors that may facilitate attacks;
  - To identify critical system assets to be protected to avoid attacks.
- **The existence of an actuator-attack resilient supervisor is open.**
- **The synthesis complexity is high.**

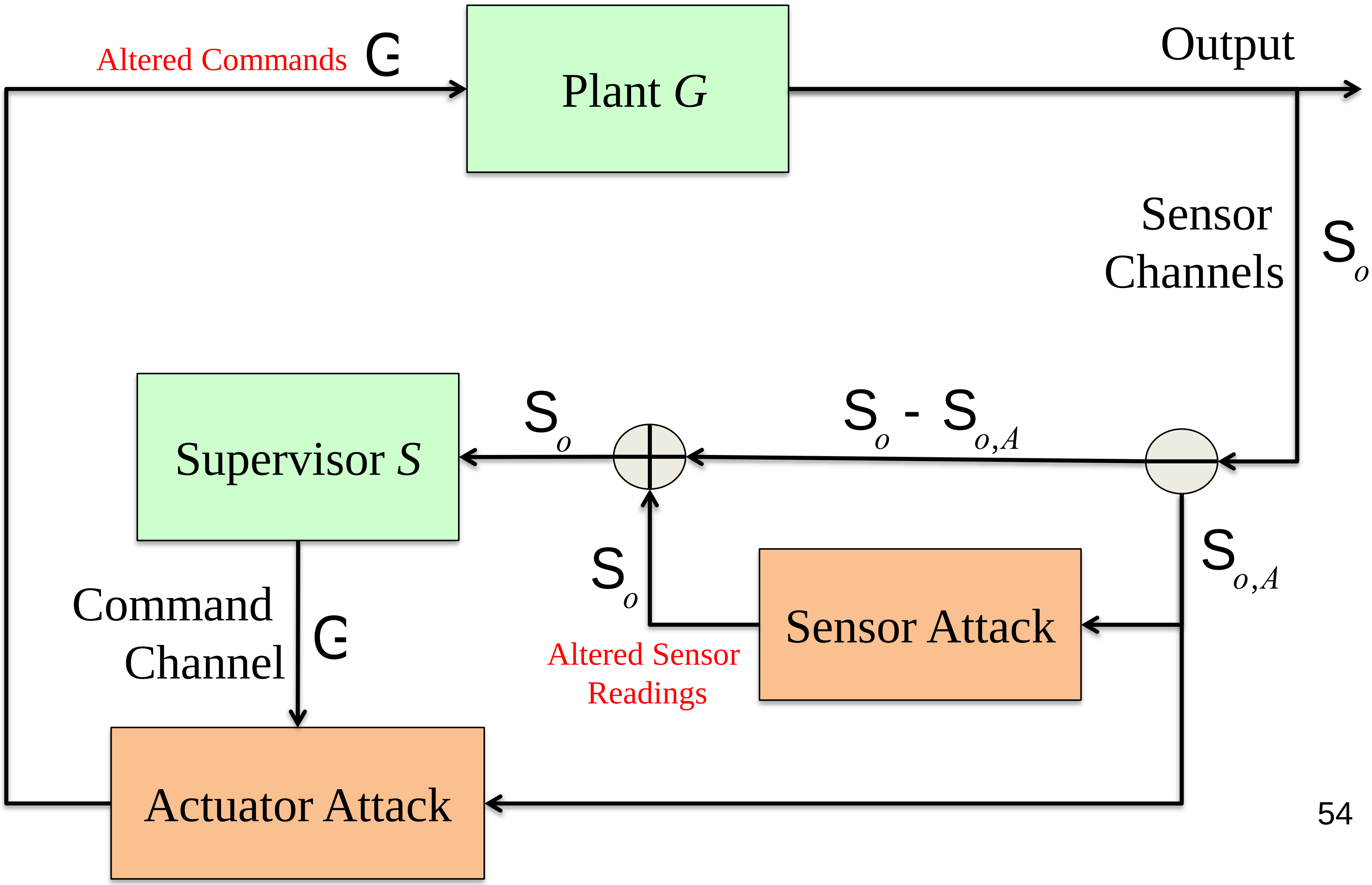
Future Works

- To improve modeling expressiveness for more types of attacks.
- To consider a unified framework for sensor and actuator attacks[12][13].
- To explore new attack resilient control strategies.
- To facilitate data-driven learning of (G, S) and A.
- Finally, to apply theory to realistic industrial application.

[12] L. Lin, R. Su. Synthesis of covert actuator and sensor attacks as supervisor synthesis. *15th IFAC WODES*, accepted, Rio de Janeiro, 2020.

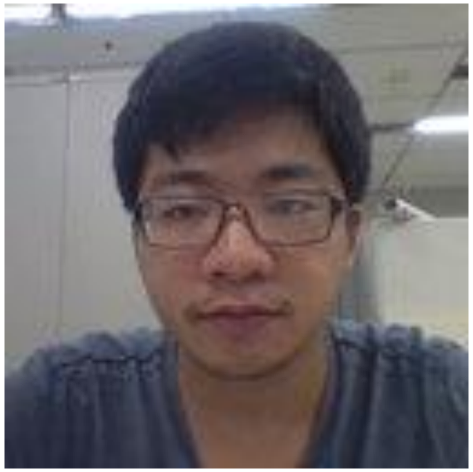
[13] L. Lin, R. Su. Synthesis of covert actuator and sensor attackers. *Automatica*, accepted, 2021.

A Holistic Cyber Security Framework



Acknowledgement

- Team members


Lin Liyong (Postdoc)

Zhu Yuting (PhD Student)

Tai Ruochen (PhD Student)

Ministry of Education
SINGAPORE

NATIONAL RESEARCH FOUNDATION
PRIME MINISTER'S OFFICE
SINGAPORE

DELTA

感谢大家!

Thank you!