Automatica 94 (2018) 35–44

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Supervisor synthesis to thwart cyber attack with bounded sensor reading alterations[☆]

Rong Su

School of Electrical & Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore, 639798, Singapore

ARTICLE INFO

Article history:
Received 16 August 2016
Received in revised form 12 January 2018
Accepted 22 March 2018

Keywords:
Discrete-event systems
Supervisory control
Cyber security
Attack under bounded sensor reading alterations
Partial observation
Controllability

ABSTRACT

One of the major challenges about cyber–physical systems is how to protect system integrity from cyber attacks. There has been a large number of different types of attacks discussed in the literature. In this paper we aim to investigate one special type of attacks in the discrete-event system framework, where an attacker can arbitrarily alter sensor readings after intercepting them from a target system, aiming to trick a given supervisor to issue improper control commands, which can drive the system to an undesirable state. We first consider the cyber attack problem from an attacker's point of view, and formulate an attack-with-bounded-sensor-reading-alterations (ABSRA) problem. We then show that the supremal (or least restrictive) ABSRA exists and can be computed, as long as the plant model and the supervisor model are regular, i.e., representable by finite-state automata. Upon the synthesis of the supremal ABSRA, we present a synthesis algorithm, which computes a supervisor that is ABSRA-robust in the sense that any ABSRA will either be detectable or inflict no damage to the system.

1. Introduction

A cyber–physical system (CPS) is a mechanism controlled or monitored by computer-based algorithms. Examples of CPS include smart grid, autonomous automobile systems, medical monitoring, process control systems, distributed robotics, and automatic pilot avionics, etc. The connection between the cyber part and the physical part heavily relies on sensor and communication networks, which has been raising a major security concern, as different types of cyber attacks can tamper the data collection processes and interfere safety critical decision making processes, which may cause irreparable damages to the physical systems being controlled and to people who depend on those systems (Cherdantseva et al., 2016; Evancich & Li, 2016).

There has been a growing number of publications addressing the cyber security issues from both the computer science community, which focuses on the computer computation related issues, and the systems control community, which focuses on issues related to the system dynamics affected by cyber attacks. Recently, more and more efforts have been made in classifying different types of malicious attacks, assuming that the attackers are sufficiently intelligent (Cardenas, Amin, & Sastry, 2008; Fawzi, Tabuada, & Diggavi, 2014; Knowles, Prince, Hutchison, Disso, & Jones, 2015; Teixeira, Perez, Sandberg, & Johansson, 2012), instead of merely just generating random failures, which is well studied in the fields of reliability and fault tolerant control. Typically, an intelligent attacker requires *system knowledge*, and abilities for *resource disclosure* and *resource disruption* in order to carry out a successful attack, which is covert to a system user until the attacker's goal of causing a damage to the system is achieved. So *covertness* and *damage infliction* are two major characteristics of a successful attack. By analysing different intelligent cyber attacks, proper countermeasures may be developed to prevent a target system from being harmed by a specific type of attacks.

In this paper we study a special type of data deception attacks in the discrete-event system framework, where an attacker can intercept sensor measurements (or observations) modelled by observable events and alter them arbitrarily but with an upper bound imposed on the length of each altered observation sequence. By sending those altered observation sequences to a given supervisor, whose function is known to the attacker in advance, the attacker can deliberately and covertly guide the system to move into some undesirable states without making any change to the supervisor. The key challenge is how to "fool" the supervisor to make it believe that the system is operating correctly, while using the supervisor's own control functions to carry out the attack, i.e., to lead the system move into a bad state. To this end, we first propose a novel concept of *attackability* and the concept of *attack under bounded sensor*

☆ The support from Singapore Ministry of Education Tier 1 Academic Research Grant M4011221.040 RG84/13 (2013-T1-002-177) and from Singapore National Research Foundation Delta-NTU Corporate Lab Program (SMA-RP2) are gratefully acknowledged. The material in this paper was partially presented at the 2017 American Control Conference, May 24–26, 2017, Seattle, WA, USA. This paper was recommended for publication in revised form by Associate Editor Christoforos Hadjicostis under the direction of Editor Christos G. Cassandras.

E-mail address: rsu@ntu.edu.sg.

https://doi.org/10.1016/j.automatica.2018.04.006
0005-1098/© 2018 Elsevier Ltd. All rights reserved.

reading alterations (ABSRA), which can be modelled as a finite-state automaton, possessing the properties of covertness, damage infliction and control feasibility under partial observations. Then we show that the supremal (or least restrictive) ABSRA exists and is computable via a specific synthesis algorithm, as long as both the plant model G and the given supervisor S are finitely representable. Upon this novel ABSRA synthesis algorithm, we present a supervisor synthesis algorithm, which can ensure that a nonempty synthesized supervisor will be "robust" to any ABSRA, in the sense that such an attack will either reveal itself to the supervisor owing to abnormal event executions (so that contingent actions can be taken by the supervisor, which is outside the scope of this paper) or will not be able to bring the system to a bad state (i.e., no damage will be inflicted).

Our construction of an ABSRA model as a finite-state automaton is inspired by recent works on opacity analysis and enforcement (Dubreil, Darondeau, & Marchand, 2010; Saboori & Hadjicostis, 2007, 2013; Wu & Lafortune, 2014), which aim to analyse and/or enforce (via observable event insertions) the capability of a system to prevent a potential attacker from correctly determining the actual state of the system. A comprehensive survey on this subject can be found in Jacob, Lesage, and Faure (2016). Owing to different objectives of two frameworks, the modelling details and synthesis algorithms are different. There are some works on cyber attack detection and prevention in the discrete-event community (Carvalho, Wu, Kwong, & Lafortune, 2016; Paoli, Sartini, & Lafortune, 2011; Thorsley & Teneketzis, 2006), mainly from an adaptive fault tolerant control point of view, which heavily rely on real-time fault diagnosis to identify the existence of an attack and then take necessary supervisory control actions. In Rasouli, Miehling, and Teneketzis (2014) the authors present a supervisory control approach for a dynamic cyber-security problem that captures progressive attacks to a computer network, which aims to compute an optimal policy in a game theoretical setup. In those works the intelligence of an attacker is not explicitly modelled, and an attack is treated as a fault or an (unintelligent) opponent. As a contrast, we do not rely on real time attack detection, but on prior knowledge of attack models, which assume that an attacker is intelligent to deliver attacks covertly and effectively, as captured by the concept of attackability, and simply build attack-robustness features into a supervisor to ensure that the supervisor will not be affected by any ABSRA unnoticeably. It is this robust control nature distinguishes our works from existing DES-based cyber attack detection and prevention approaches, which fall in the adaptive control domain. Our ABSRA-robust supervisor synthesis bears a slight conceptual similarity to the problem of supervisory control with intermittent sensor failures (Alves, Basilio, da Cunha, Carvalho, & Moreira, 2014; Rohloff, 2012), although the problem setups and solutions are completely different.

The remainder of the paper is organized as follows. In Section 2 we review the basic concepts and operations of discrete event systems. Then we formulate an ABSRA synthesis problem in Section 3, where we show that the supremal ABSRA exists and computable. In Section 4 we present an algorithm to synthesize an ABSRA-robust supervisor, and use a toy example to show that the supremal ABSRA-robust supervisor usually does not exist. A simple yet realistic example runs through the entire paper to illustrate all relevant concepts and algorithms. Conclusions are drawn in Section 5.

2. An ABSRA problem

In this section we first recall basic concepts used in the Ramadge–Wonham supervisory control paradigm. Then we introduce the concept of ABSRA, followed by a concrete ABSRA synthesis algorithm, which reveals that the supremal ABSRA is computable, as long as both the plant model and the given supervisor are regular.

2.1. Preliminaries on supervisory control

Given a finite alphabet Σ, let Σ^* be the free monoid over Σ with the empty string ϵ being the unit element and the string concatenation being the monoid operation. Given two strings $s, t \in \Sigma^*$, we say s is a *prefix substring* of t, written as $s \le t$, if there exists $u \in \Sigma^*$ such that $su = t$, where su denotes the concatenation of s and u. Any subset $L \subseteq \Sigma^*$ is called a *language*. The *prefix closure* of L is defined as $\overline{L} = \{s \in \Sigma^* | (\exists t \in L) s \le t\} \subseteq \Sigma^*$. Given two languages $L, L' \subseteq \Sigma^*$, let $LL' := \{ss' \in \Sigma^* | s \in L \wedge s' \in L'\}$ denote the concatenation of two sets. Let $\Sigma' \subseteq \Sigma$. A mapping $P : \Sigma^* \to \Sigma'^*$ is called the *natural projection* with respect to (Σ, Σ'), if

(1) $P(\epsilon) = \epsilon$,
(2) $(\forall \sigma \in \Sigma) P(\sigma) := \begin{cases} \sigma & \text{if } \sigma \in \Sigma', \\ \epsilon & \text{otherwise,} \end{cases}$
(3) $(\forall s\sigma \in \Sigma^*) P(s\sigma) = P(s)P(\sigma)$.

Given a language $L \subseteq \Sigma^*$, $P(L) := \{P(s) \in \Sigma'^* | s \in L\}$. The inverse image mapping of P is

$$P^{-1} : 2^{\Sigma'^*} \to 2^{\Sigma^*} : L \mapsto P^{-1}(L) := \{s \in \Sigma^* | P(s) \in L\}.$$

A given target plant is modelled as a *deterministic finite-state automaton*, $G = (X, \Sigma, \xi, x_0, X_m)$, where X stands for the state set, Σ for the alphabet, $\xi : X \times \Sigma \to X$ for the (partial) transition function, x_0 for the initial state and $X_m \subseteq X$ for the marker state set. We follow the notation in Wonham (2014), and use $\xi(x, \sigma)!$ to denote that the transition $\xi(x, \sigma)$ is defined. For each state $x \in X$, let $En_G(x) := \{\sigma \in \Sigma | \xi(x, \sigma)!\}$ be the set of events enabled at x in G. The domain of ξ can be extended to $X \times \Sigma^*$, where $\xi(x, \epsilon) = x$ for all $x \in X$, and $\xi(x, s\sigma) := \xi(\xi(x, s), \sigma)$. The *closed* behaviour of G is defined as $L(G) := \{s \in \Sigma^* | \xi(x_0, s)!\}$, and the *marked* behaviour of G is $L_m(G) := \{s \in L(G) | \xi(x_0, s) \in X_m\}$. G is *nonblocking* if $\overline{L_m(G)} = L(G)$. We will use $\mathbb{N}$ to denote natural numbers, $|G|$ for the size of its state set, and $|\Sigma|$ for the size of Σ. Given two finite-state automata $G_i = (X_i, \Sigma, \xi_i, x_{i,0}, X_{i,m})$ $(i = 1, 2)$, the *meet* of G_1 and G_2, denoted as $G_1 \wedge G_2$, is a (reachable) finite-state automaton whose alphabet is Σ such that $L(G_1 \wedge G_2) = L(G_1) \cap L(G_2)$ and $L_m(G_1 \wedge G_2) = L_m(G_1) \cap L_m(G_2)$.

We now recall the concept of supervisors. Let $\Sigma = \Sigma_c \dot\cup \Sigma_{uc} = \Sigma_o \dot\cup \Sigma_{uo}$, where Σ_c (Σ_o) and Σ_{uc} (Σ_{uo}) are disjoint, denoting respectively the sets of *controllable* (*observable*) and *uncontrollable* (*unobservable*) events. A *(feasible) supervisory control map of* G *under partial observation* $P_o : \Sigma^* \to \Sigma_o^*$ is defined as $V : L(G) \to 2^{\Sigma}$, where

- $(\forall s \in L(G))(\forall \sigma \in \Sigma_{uc}) s\sigma \in L(G) \Rightarrow \sigma \in V(s)$,
- $(\forall s, s' \in L(G)) P_o(s) = P_o(s') \Rightarrow V(s) = V(s')$.

For each $s \in L(G)$, $V(s)$ is interpreted as the set of events allowed to be fired after s. Thus, a supervisory control map will not disable any uncontrollable events, and will impose the same control pattern after strings, which cannot be distinguished based on observations. Let V/G denote the closed-loop system of G under supervision of V, i.e.,

- $\epsilon \in L(V/G)$,
- $(\forall s \in L(V/G))(\forall \sigma \in \Sigma) s\sigma \in L(V/G) \iff s\sigma \in L(G) \wedge \sigma \in V(s)$,
- $L_m(V/G) := L_m(G) \cap L(V/G)$.
- $L(V/G) = \overline{L_m(V/G)}$.

The control map V is *finitely representable* if V/G can be described by a finite-state automaton, say $S = (Z, \Sigma, \delta, z_o, Z_m = Z)$, such that