Computers & Security 137 (2024) 103593

Contents lists available at ScienceDirect

Computers & Security

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

Identifying vulnerabilities of industrial control systems using evolutionary multiobjective optimisation

Nilufer Tuptuk [a,*], Stephen Hailes [b]

[a] Department of Security and Crime Science, University College London, London, United Kingdom
[b] Department of Computer Science, University College London, London, United Kingdom

ARTICLE INFO

Keywords:
Cybersecurity
Vulnerabilities
Adversarial attacks
Evolutionary multiobjective optimisation
Industrial control systems

ABSTRACT

In this paper, we propose a novel methodology to assist in identifying vulnerabilities in real-world complex heterogeneous industrial control systems (ICS) using two Evolutionary Multiobjective Optimisation (EMO) algorithms, NSGA-II and SPEA2. Our approach is evaluated on a well-known benchmark chemical plant simulator, the Tennessee Eastman (TE) process model. We identified vulnerabilities in individual components of the TE model and then made use of these vulnerabilities to generate combinatorial attacks. The generated attacks were aimed at compromising the safety of the system and inflicting economic loss. Results were compared against random attacks, and the performance of the EMO algorithms was evaluated using hypervolume, spread, and inverted generational distance (IGD) metrics. A defence against these attacks in the form of a novel intrusion detection system was developed, using machine learning algorithms. The designed approach was further tested against the developed detection methods. The obtained results demonstrate that the developed EMO approach is a promising tool in the identification of the vulnerable components of ICS, and weaknesses of any existing detection systems in place to protect the system. The proposed approach can serve as a proactive defense tool for control and security engineers to identify and prioritise vulnerabilities in the system. The approach can be employed to design resilient control strategies and test the effectiveness of security mechanisms, both in the design stage and during the operational phase of the system.

1. Introduction

Industrial Control Systems (ICS) are command and control systems that are found at the core of the national critical infrastructure services such as gas; electricity; oil; water supply; telecommunication; transportation; process manufacturing (chemicals, pharmaceuticals, paper, food and beverages, and other batched-based manufacturers); and discrete manufacturing (automobiles, ships, computers and many other durable goods). The security of ICS is of critical importance in industrialised economies: they are so pervasive that national security, public health and safety, and economic growth all rely on their correct operation. In the past, the security of ICS was achieved simply through isolation and controlling physical access. However, ICS are making increasing use of network technologies, commercial-off-the-shelf (COTS) components, and wireless systems driven by advantages such as low-cost, increased sensing, and communication capacity and convenience.

With these technological advances, factory and plant networks are evolving into highly interconnected systems running over multiple layers. In these settings, signals sent between control components (i.e. sensors, controllers, and actuators), are transmitted through a shared network, commonly known as a networked control system (Patton et al., 2007). The number of highly motivated and skilled adversaries capable of executing sophisticated attacks against networked control systems is on the rise. Some of the past attacks include the attack against the operational systems of Evraz Steel in North America (CBC, 2020); the attack on Ukraine's Power Grid (Dragos Inc., 2017) that targeted the electric transmission system in Kiev; the attack against a German Steel Mill (BSI, 2014) that caused unspecified but "massive" physical damage; malware attacks such as Duqu (Symantec, 2011a) and Havex (F-Secure, 2014) that targeted ICS for industrial espionage; and Stuxnet (Symantec, 2011b) that targeted Iran's Natanz nuclear plant, and destroyed centrifuges installed at the time of the attack. As the evidence from these attacks shows the potential outcome of a successful attack on a critical service ranges from injuries and fatalities, through serious damage to the environment, to catastrophic nationwide economic loss due to production losses or degradation of products and services. Disrupt-

* Corresponding author.
E-mail addresses: n.tuptuk@ucl.ac.uk (N. Tuptuk), s.hailes@ucl.ac.uk (S. Hailes).

https://doi.org/10.1016/j.cose.2023.103593
Received 12 April 2023; Received in revised form 1 November 2023; Accepted 6 November 2023
Available online 14 November 2023

ing the availability of these systems or denying access, even for a short time, can lead to substantial harm to people and may impact public confidence, causing a widespread sense of insecurity.

Despite the technological advances in ICS, there is a notable research gap in building security tools designed to proactively search for vulnerabilities related to the process level components (such as sensors, actuators and controllers) and leverage this knowledge to make informed decisions about security. It is essential for national economic resilience that we explore better ways of searching for vulnerabilities in the industrial processes and defence mechanisms, and understand the impact of these vulnerabilities would be if they were to be exploited. Considerable research (Cárdenas et al., 2011; Huang et al., 2009; Krotofil and Cárdenas, 2013; Genge et al., 2012; Wang et al., 2018; Di Pietro et al., 2013; Huang et al., 2018) has focused on developing threat models and analysing a variety of attacks, aimed at modifying process measurements (sensor measurements) or manipulated variables (control actions sent by the controller to actuators), or manipulating the control algorithm (e.g. set points), however, there is a lack of research on automating this process and investigating the impact of combinatorial attacks involving multiple simultaneous attacks, and testing the effectiveness of existing security countermeasures.

In this paper, we present an approach to optimise attacks to identify the weakest components within a ICS. To illustrate this approach, we search for process level attacks that have an impact, potentially damage the safety of ICS and the economics of the production, using the least effort and attacks that have the least likelihood of detection. To achieve this, we establish the problem as a multiobjective optimisation problem and investigate the effectiveness of Evolutionary Multiobjective Optimisation (EMO) algorithms to generate effective and optimal attacks. The approach is designed to be used by defenders of the systems, control and security engineers, to analyse the attack surface of an ICS as well as evaluating the impact of attacks and the robustness of the system and its security components. Consequently, the designers and operators of the systems can use the approach as address the weaknesses and develop resilient control systems.

The remainder of the paper is organised as follows. Section 2 presents the related work from the literature. Section 3 presents an overview of the background material and assumptions related to our approach. This includes the characteristics of the case study, the Tennessee Eastman (TE) process model; a description of the multiobjective optimisation problems; and assumptions related to the threat model and adversary. Section 4 covers the methodology including details of the detection methods used to evolve attacks against detection; EMO algorithms and the performance metrics used for comparing EMO algorithms; and the random approach used for generating combinatorial attacks, to compare with the EMO approach. Section 5 presents experimental results. In section 6, the application of results are discussed. Conclusion and future work are presented in Section 7.

2. Related work

Multiobjective optimisation (MOO) has been widely applied to real-world complex scientific and engineering challenges including in the realm of industrial processes (Cerda-Flores et al., 2022; Rangaiah et al., 2020; Liu et al., 2020), from optimising industrial workflows to designing control strategies. However, despite its widespread in control engineering, there appears to be a gap in the application of EMO algorithms for improving the security of cyber-physical systems, including Industrial Control Systems (ICS). On the other hand, evolutionary algorithms have been extensively used to improve the security mechanisms used within IT networks in a wide range of applications. Researchers have employed Genetic Algorithm (GA) (Li, 2004; Xia et al., 2005; Vollmer et al., 2011; Ojugo et al., 2012; Hoque et al., 2012; Diaz-Gomez et al., 2005; Goyal and Kumar, 2008), and Genetic Programming (GP) (Wei and Traore, 2004; Pastrana et al., 2011) to evolve new rules to detect new forms of network intrusion. Our previous work Mrugala et al. (2016) Mrugala et al. (2017) used GP to attack a wireless sensor network (WSN) protected by an artificial immune intrusion detection system. Kayacik et al. (2006) used GP to evolve variants of buffer overflow attacks against an open-source signature-based intrusion detection system. John et al. (2014) applied GA to the improvement of moving target defence, in which the defence changes the system's attack surface to disrupt the intelligence gathered by the attacker. Dewri et al. (2007) used GA with multiobjective optimisation to investigate optimal security measures for a system. Garcia and Erb Lugo (2017), Hemberg et al. (2018), Rush et al. (2015) used co-evolution to model attacker and defence dynamics for network security. Co-evolutionary concepts have also been investigated to prevent faults and cascading blackouts in electric power transmission systems (Service et al., 2007; Service and Tauritz, 2009); automate red teaming for military scenarios (Decraene et al., 2010); and improve the performance of malware detectors (Bronfman-Nadas et al., 2018).

In recent years, a new and rapidly growing field known as Adversarial Machine Learning (AML) (Huang et al., 2011) has emerged, dedicated to identifying and exploiting vulnerabilities in Machine Learning (ML) models. Within AML, researchers have identified two main classes of adversarial attacks that exploit the vulnerabilities of machine learning systems: *evasion attacks* and *poisoning attacks*. Evasion attacks are carried out during the testing phase of ML models by making small subtle changes to input data to make the model produce an incorrect output. Evasion attacks have been used to generate attacks against anomaly-based detection models in ICS (Erba et al., 2019). Poisoning attacks manipulate the training phase of the ML models by injecting malicious data points or biases into the training dataset. These attacks are designed to compromise the model's integrity, leading to incorrect predictions when the model is deployed, for example, a particular cyber-attack is not detected when the model is operational (Kravchik et al., 2021).

Our approach makes no assumptions related to the intrusion detection models or mechanisms, and the training methods. It complements existing studies in applying evolutionary multiobjective optimisation to evolve combinatorial attacks to uncover vulnerabilities within the system and in its attack detection mechanisms. We investigate both the worst-case condition where ICS has no security protection, and also where there are some measures against attacks, in the form of a novel intrusion detection system.

3. Background knowledge and assumptions

This section explains the essential background knowledge required to understand the methodology. It covers the case study used for experimental work, objectives of the multiobjective optimisations and threat modelling assumptions.

3.1. Case study: the Tennessee Eastman (TE) process

To develop and explore the effectiveness of evolutionary multiobjective optimisation as a possible candidate for identifying vulnerabilities in ICS, a well-known chemical plant, the Tennessee Eastman (TE) process control model (Downs and Vogel, 1993) was selected.

Our reasons for selecting this process are: i) it is a well-known model that has been widely studied; ii) it is a complex, highly non-linear system with a number of components that reflect a real ICS process; iii) safety and economic viability can be quantified; iv) the code and model is available, and have been revised and validated over the years; and v) it continues to be a relevant model for both the control and more recently, the security communities. We are not aware of any other open-source model that has these properties. The TE model is based on a real chemical process; however, the identities of the reactants and products were hidden to maintain commercial confidentiality. The process has eight components: four gaseous reactants (A, C, D, E), two products (G,

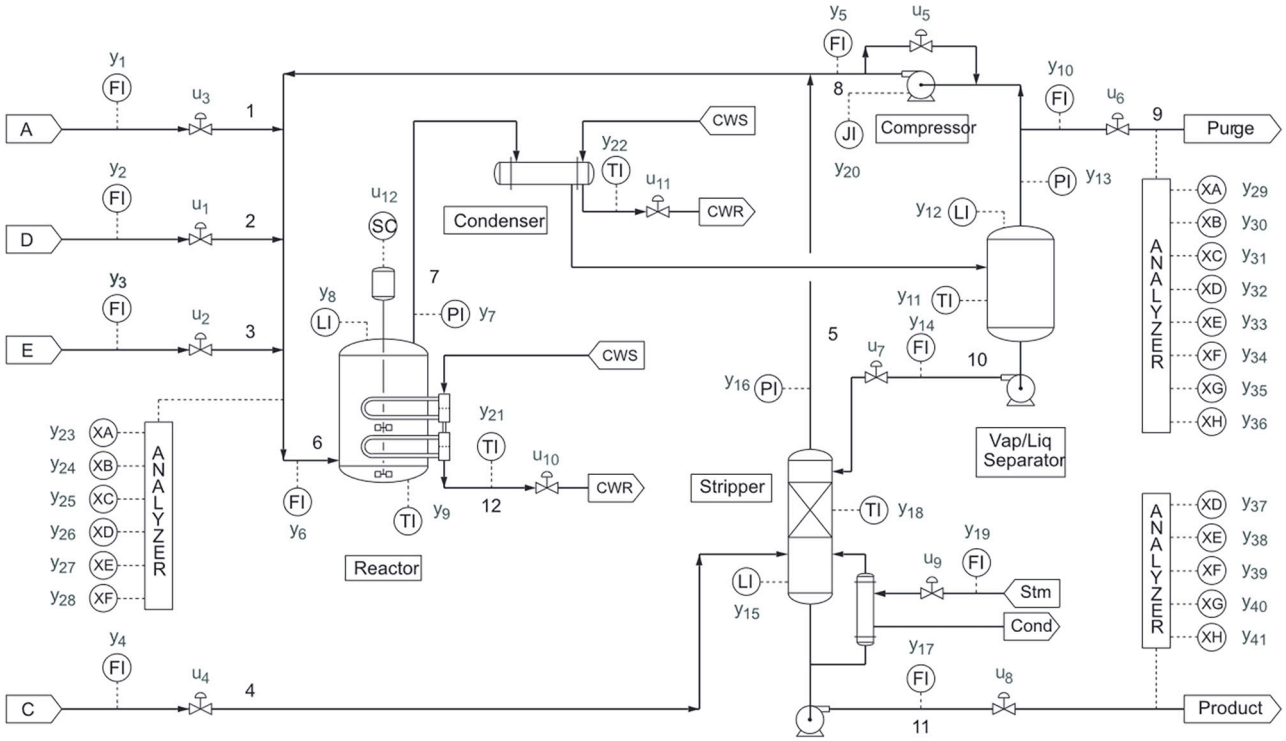


Fig. 1. Tennessee Eastman challenge model (Downs and Vogel, 1993, image taken from Glavan et al., 2013).

H), an inert component (B) and a by-product (F). These reactions are (Downs and Vogel, 1993):

A(g) + C(g) + D (g) → $G(liq)$, Product 1,
A(g) + C (g) + E (g) → $H(liq)$, Product 2,
A(g) + E (g) →F(liq), Byproduct,
3D(g) → $2F(liq)$, Byproduct.

The process is illustrated in Fig. 1, and consists of five major components Downs and Vogel (1993): a reactor, a product condenser, a vapour-liquid separator, a recycle compressor and a product stripper.

There are 41 process measurement variables known as XMEASs (sensors, denoted as y signals) and 12 manipulated variables known as XMVs (valves/actuators, denoted as u signals), illustrated in Fig. 1, that are involved in controlling and monitoring the plant. Later we will be evolving attacks against these measurement and manipulated variables. The main control objectives of the plant are (Downs and Vogel, 1993): to maintain the process variables at the desired values; to ensure that the operational conditions are within the equipment constraints; and to minimise variability of the product rate and product quality during disturbances. To protect the safety of the process, the TE model has a set of operating constraints that are known as normal operating limits and shutdown operating limits. If the process reaches the shutdown limits, it automatically shuts the plant down. These constraints (such as low/high limits of reactor pressure, reactor level, reactor temperature, product separator level and stripper base level) (Downs and Vogel, 1993) are established to protect the personnel, equipment, production, and comply with regulatory requirements. The operating costs of the TE process are calculated according to the following equation (Downs and Vogel, 1993):

total costs =(purge costs)(purge rate)
+ (product stream costs)(product rate)
+ (compressor costs)(compressor work)
+ (steam costs)(steam rate)

The TE process problem makes no recommendation as to what needs to be controlled and leaves the selection of controlled variables and control strategies to the control engineers. Most proposed solutions do not control all the variables. The control strategy used in this paper is that described by Larsson et al. in Larsson et al. (2001) using 16 process measurements and 9 manipulated variables. The process model used in this paper is developed by Ricker, available from his home page Ricker (1998). The code is implemented in C, with a MATLAB/Simulink interface via an S-function implementation. Isakov and Krotofil (2015) extended the original Simulink model by enhancing it with Simulink blocks that enable integrity and denial-of-service (DoS) attacks to be carried out on the sensors and manipulated variables. We extended their model with replay attacks and made further small changes needed to carry out the work in this paper.

3.2. Multiobjective optimisation

In many real-world problems, decisions need to be made on the basis of multiple competing or conflicting objectives and constraints. This is often the case when making decisions related to cybersecurity investment or security hardening. It involves balancing the risk of attacks against a limited budget allocated for purchasing defensive countermeasures. In such situations, formulating the problem as a multiobjective optimisation with multiple choices can help to determine the trade-offs among the objectives in a more effective manner (Riquelme et al., 2015; Fielder et al., 2016; Dewri et al., 2007). These approaches search for the set of *non-dominated* or *Pareto-optimal* solutions. A solution is defined as non-dominated if there are no other solutions that will improve an objective without degrading at least one other objective (Deb et al., 2002). Once the set of Pareto-optimal solutions, has been identified, a decision maker can make a decision by examining the trade-offs represented by individual solutions within the set. MOO takes a problem with multiple objectives and simultaneously seeks to optimise all objectives, providing solutions in, or close to, the true Pareto-optimal set. More often than not, this is an estimate because determining the true Pareto-optimal set for real-world problems is hard, either because the search space is too

large or time and computation resources required to find solutions are expensive.

In this study, we are concerned with finding optimised solutions for the following multiobjective problems:

1. **Compromising the safety of the plant (shutting down the plant)**: Minimise plant operating time, and minimise the effort required to carry out attacks.
2. **Causing economic loss:** Maximise the operating cost of the plant, and minimise the effort required to carry out attacks.
3. **Evade detection while causing economic loss:** Maximise operating cost of the plant, minimise detection (alarm) probability, and minimise the effort required to carry out attacks.

The generation of optimal attacks against industrial control systems with a large number of components involves the selection of many parameters: attack targets (controllers, sensors, actuators); attack types; and attack parameters for these attacks (e.g. mode of attacks, attack start times and attack duration). In practice identifying the true Pareto-optimal set to such problems may not always be achievable, however, evolutionary multiobjective optimisation stands as a promising approach to identify an estimate of best trade-off attacks in such complex systems at a reasonable computational cost.

3.3. Threat modelling assumptions

Fig. 2 shows our underlying threat model that is based on common attacks against networked systems. The adversary is capable of intercepting communication from the sensor to the controller (process measurements), and controller to the actuator (manipulated variables). The attacks we consider are categorised as DoS, integrity (man-in-the-middle) and replay attacks. We will investigate the impact of these attacks in terms of measuring the impact on the safety and operating costs of the plant. In the following section, we briefly discuss what this means, and explain how the attacks are modelled.

Past studies have investigated the safety and economic impact of DoS and man-in-the-middle attacks on the TE model (Huang et al., 2009; Cárdenas et al., 2011; Krotofil and Cárdenas, 2013). Building on their attack model, our focus is to generate optimised combinatorial attacks. We extend their analysis by undertaking a more comprehensive search and examining the possibility of forming combinations of attacks. The attack parameters are as follows:

Attack Targets: The control strategy selected for our investigation, Larsson et al. in (2001), uses 16 XMEAS variables, and 9 XMV variables. An adversary may attempt to manipulate signals that are sent from XMEASs to controllers (process measurements), and/or from controllers to the XMVs (manipulated variables). The process run time used in this study is 72 hours. An attack begins at time t_s and ends at t_e, it can start any time, between the start of the plant and the end, $t \in [0-72]$. Let I_a be the attack interval, let y_i be the output of sensor i at time t, and let u_i be the controller signal i at time t. The manipulated control (XMV) signal $u_i^a(t)$ and process measurement (XMEAS) $y_i^a(t)$ are as follows:

$$y_i^a(t) = \begin{cases} y_i^{(t)}, for\ t \notin I_a \\ \hat{y}_i^{(t)}, for\ t \in I_a \end{cases} \tag{1}$$

$$u_i^a(t) = \begin{cases} u_i^{(t)}, for\ t \notin I_a \\ \hat{u}_i^{(t)}, for\ t \in I_a \end{cases} \tag{2}$$

where $\hat{y}_i^{(t)}$ and $\hat{u}_i^{(t)}$ are the modified values the adversary sends.

To investigate the impact of attacks, we considered three types of attacks for this study: *DoS*, *integrity* and *replay* attacks.

A **DoS attack**, is an interruption attack in which a signal is not received by its intended destination. For example, let y_i be the output of sensor i at time t, and let u_i be the output from the controller to actuator signal i at time t. When the attack occurs, the response strategy

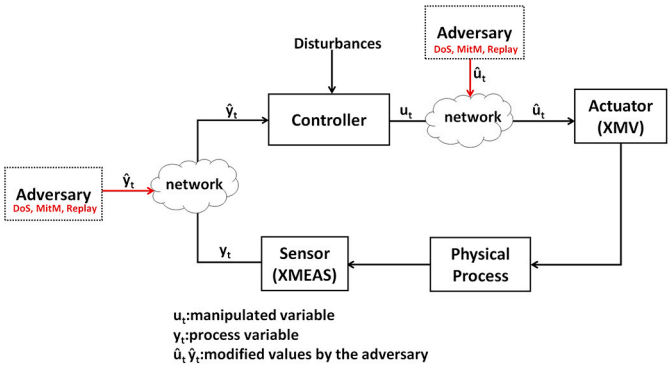


Fig. 2. ICS attack model against the networked control system.

for the controller or the actuator is to use the last received value as the current reading:

$$\begin{aligned} \hat{y}_i^{(t)} &= y_i(t_{s-1}) \\ \hat{u}_i^{(t)} &= u_i(t_{s-1}) \end{aligned} \tag{3}$$

where t_s is the attack start time, $\hat{y}_i^{(t)}$ and $\hat{u}_i^{(t)}$ are the modified values the adversary sends.

An **integrity attack** involves an attacker manipulating signals by changing their values. A naïve attacker may listen to the transmitted values and modify them so that they are still within the ranges of possible plant values since this has the potential to cause some damage. One way to achieve this is to try to modify the sensor measurements (y_i) and manipulated variables (controller signals) (u_i) using observed upper minimum $(integrity_{min})$ and lower maximum $(integrity_{max})$ values:

An $integrity_{min}$ is where the actual output of the sensor or controller signal i at time t is replaced with a minimum value:

$$\begin{aligned} \hat{y}_i^{(t)} &= \underset{t \in T}{min}(y_i(t)) \\ \hat{u}_i^{(t)} &= \underset{t \in T}{min}(u_i(t)) \end{aligned} \tag{4}$$

An $integrity_{max}$ is where the actual output of the sensor or controller signal i at time t is replaced with a maximum value:

$$\begin{aligned} \hat{y}_i^{(t)} &= \underset{t \in T}{max}(y_i(t)) \\ \hat{u}_i^{(t)} &= \underset{t \in T}{max}(u_i(t)) \end{aligned} \tag{5}$$

A **replay attack** involves forging sensor measurements or manipulated variables as in the integrity attack but, this time, it repeatedly replays legitimate data it observed earlier:

$$\begin{aligned} y_i^r &= [y_i^{(r_{start})}, \ldots, y_i^{(r_{end})}] \\ u_i^r &= [u_i^{(r_{start})}, \ldots, u_i^{(r_{end})}] \\ \hat{y}_i^{(t)} &= \hat{y}_i^r[t \bmod \text{len } y_i^r] \\ \hat{u}_i^{(t)} &= \hat{u}_i^r[t \bmod \text{len } u_i^r] \end{aligned} \tag{6}$$

where y_i^r and u_i^r are the signals recorded by the adversary from the replay period, r_{start} to r_{end}.

3.4. Adversary assumptions

We consider two types of adversaries: i) an adversary targeting the safety of the plant by attempting to shut it down using the least effort; and ii) an adversary targeting the operating cost of the plant to increase economic loss using the least effort. The effort of the attack is calculated as the total number of sensors and actuators that must be attacked. Often, attackers may not have the budget, skilled personnel and time, to carry out attacks that exploit all components in a plant. We make the assumption that attackers will carry out attacks that cause them the least effort (cost) and most damage to the target system. Similarly, the defence may not have the cybersecurity budget to mitigate

every vulnerability in a system. The effort objective is used to identify the potentially most vulnerable combinations of components. This knowledge can then be used to make decision related to allocation of cybersecurity budget.

When developing attacks against detection models, we make the assumption that machine learning techniques are used for detecting these attacks. We assume an adversary who has control over their attack vector and who knows the feature space used by the detection system. However, the adversary had no knowledge of the underlying detection methods. So, essentially, the detection is a black box that can be queried. An adversary queries the detection with the attack vector and obtains a detection probability. The adversary's goal is to discover attacks that cause damage whilst evading detection and spending minimal effort.

3.5. Normal operation of TE model

To establish a baseline cost for the normal operation of the TE model, 1000 independent runs were conducted without disturbances. For this, and all subsequent runs, the plant was operated in Mode 1, the most commonly used configuration in the literature (Ricker, 1995). The following operational costs were obtained:

Operating Cost ($)	Max ($)	Mean ($)
Normal	8,218	8,208

To guide the subsequent integrity attacks later $integrity_{min}$ and $integrity_{max}$, the minimum and maximum values of the XMEAS and XMV signals observed under normal operating conditions were recorded. These values are presented in Table 2 and Table 3 under the column titled Variable Name and Range.

4. Methodology

This section outlines the four phases of our experimental work: (1) single variable attacks; (2) machine learning based detection models; (3) evolutionary multiobjective optimisations approach; and (4) random generation of combinatorial attacks.

4.1. Single variable attacks

To understand the normal operation of the TE model and establish a baseline for evaluating the impact of evolved attacks, 500 attacks were launched on each of the XMEAS and XMV signals for every type of attack.

The attacks were started at hour 2 with the intention of running them for the remainder of the simulation time (i.e. 70 hours). For every attack, a new seed was employed for the random number generator to introduce a degree of random noise in the plant's operation (i.e. sensor and actuator signals). This way, even though the attacks were started at the same time, the signals targeted varied across each of the 500 runs.

4.2. Machine learning based detection models

To evolve attacks against attack detection systems, a set of commonly employed machine learning algorithms were utilised to create detection methods. The data used for training the detection models is multivariate time series data. The TE model generates a total of 53 data variables, which include 41 process measurements (XMEAS) and 12 manipulated variables (XMV). Each run of the plant produces a data matrix of size 53 x 36000 data points, corresponding to 500 points per hour over a span of 72 hours. This data was used to train the detection methods. To detect attacks, three supervised learning methods – decision tree (CART, tree depth = 50), AdaBoost (with CART, number of estimators = 100), random forest (with CART, number of estimators = 25) – and one unsupervised learning method – one-class SVM (kernel = RBF, $\upsilon = 0.00346$ and $\gamma = 0.018$) – were used. The training data for supervised learning were selected from $integrity_{min}$ and $integrity_{max}$ attacks on measured variables (XMEAS). The attack samples were generated by carrying out integrity attacks on XMEAS signals with duration from 20 minutes and 3 hours. The dataset used for training the unsupervised learning method consists of normal operational data without any attacks. The test dataset used for evaluating the detection models was unbalanced, consists of 3,456,096 data points, with 16% of the data points corresponding to instances of attack. We made the assumption that acquiring attack data for ICS can be difficult, and it is more common to work with unbalanced datasets.

Table 1
Operators and parameters for evolutionary multiobjective optimisation.

Parameters	Value
Chromosome Size	25
Representation of Genes	Integers
Number of Generations	500-1000
Parent Population (μ)	100-400
Crossover Rate ($cxpb$)	0.8-0.90
Mutation Rate ($mutpb$)	0.05-0.15
Probability of mutating a gene	0.05-0.08
EMO Algorithms	NSGA-II, SPEA2

4.3. Evolutionary multiobjective optimisation approach

Two of the well-known Pareto-based evolutionary multiobjective algorithms, the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2000, 2002) and Strength Pareto Evolutionary Algorithm 2 (SPEA2) (Zitzler et al., 2001) were chosen to develop the EMO approach aimed at evolving the following attacks defined as optimisation problems:

1. **Shutdown attacks** is defined as a two-objective optimisation problem: to minimise the time required to shut down the plant (f1) and to minimise the effort required to carry out the attack (f2). The time required to shut down the plant is defined by how long the plant continues to operate once the attack starts. The plant shutdown occurs as a result of exceeding plant operating constraints. Effort is defined as the total number of sensors and actuators being attacked.
2. **Operating cost attacks** defined as a two-objective optimisation problem: to maximise the total operating cost of the plant (f1) and to minimise the effort (f2).
3. **Attacks against detection** defined as two/three objective optimisation: to maximise economic loss (f1), to minimise detection (alarm) probability (f2), and to minimise the effort (f3).

4.3.1. Representation of individuals in EMO

In the process of developing EMOs, it is important to represent the individuals (commonly referred to as chromosomes) in a format where genetic operations like crossover and mutations can be applied.

The TE model consists of 25 sensors and actuators, which serve as potential targets for attacks. As discussed before, there are four potential attack types: Denial of Service (DoS), minimal integrity ($integrity_{min}$), maximal integrity ($integrity_{max}$), and replay ($replay$).

In this study, individuals are represented as a list of 25 integers, each position (gene) of the list corresponding to the target of the attack, specifically either a sensor or an actuator. The gene values, expressed as integers, indicate the attack type and its starting time. The initial population of these individuals is generated randomly.

4.3.2. Fitness function

The fitness of individuals is evaluated by converting individuals into MATLAB scripts, which are then executed on the TE plant. The fitness of the shutdown attacks is evaluated as the duration of plant operation post-attack (f1) and the number of variables attacked (f2). The performance of the operating cost attacks (economic loss) is evaluated based on the operating cost of the plant (f1) and the number of variables attacked (f2). The performance of the attacks against detection is evaluated based on the detection probability (f1), the increased operating cost of the plant (f2), and the number of variables attacked (f3, for 3-objective optimisation).

Data collected from TE model is tested against the detection model to determine the detection probability.

Algorithm 1: Evolutionary multiobjective optimisation algorithm for generating attacks.

```
Function NSGA-II(μ, mutp, cxpb):
    ParetoFront = [];
    μ = pop size, pop = generateRandomPop(μ);
    pop = evaluateFitness(pop);
    gen = 1;
    while gen ≤ ngens do
        offspring = selTournament(pop,μ);
        offspring = crossover(offspring,cxpb);
        offspring = mutate(offspring,mutp) ;
        offspring = evaluateFitness(offspring);
        pop = selectNSGA2(offspring+ pop,μ);
        ParetoFront.update(pop);
        gen = gen + 1;
    end
    return ParetoFront;
Function SPEA2(μ, mutp, cxpb):
    ParetoFront = [];
    μ = pop size, pop = generateRandomPop(μ);
    pop = evaluateFitness(pop);
    gen = 1;
    while gen ≤ ngens do
        offspring = vary(pop,μ,mutpb,cxpb);
        offspring = evaluateFitness(offspring);
        pop = selectSPEA2(offspring+ pop,μ);
        ParetoFront.update(pop);
        gen = gen + 1;
    end
    return ParetoFront;
Function vary(pop, μ, mutp, cxpb):
    offspring = [];
    for i = 0 to μ do
        random = randomGenerator(0,1) ;
        if random < cxpb then
            ind1,ind2 = selectTwoParents(pop);
            child1,child2 = crossOver(ind1,ind2);
            offspring.add(child1) ;
        else if random < cxpb+mutp then
            ind = selectOneParent(pop) ;
            child = mutate(ind);
            offspring.add(child) ;
        else
            ind = selectOneParent(pop) ;
            offspring.add(ind) ;
        end
    end
    return offspring;
```

4.3.3. Evolution

Table 1 shows the genetic parameters and the operators used in our experimental setup. Once the fitness of the initial population$_0$ has been evaluated, the evolutionary loop begins to generate the next generation (gen = 1) of individuals, as illustrated in Algorithm 1. First, the individuals in population$_0$ are subject to the genetic variation operators to generate the subsequent generation of offspring. For NSGA-II we used the same genetic variation operators as used in the original algorithm: tournament selection, two-point crossover, and uniform mutation. Crossover and mutation rates were manually tuned based on values that are usually chosen within the literature: cross-over probabilities used are between 0.8-0.9 and mutation probability between 0.05-0.15. For SPEA2, we used the *vary* function shown in Algorithm 1, where, on each of the μ iterations, randomly picked individuals are subject to one of the three operations: two-point crossover, uniform mutation, or reproduction. Our initial experiments showed these operators performed better than the evolutionary operators that were used in standard SPEA2 (Zitzler et al., 2001): binary tournament selection, single-point crossover, and bit-flip mutation. Both NSGA-II and SPEA2's selection operators use the (parent+offspring) population to select the next generation: in other words the next generation of the population is produced from both the generated offspring and current parent population$_0$.

NSGA-II creates a Pareto rank of individuals from (parent+offspring) population using non-dominated sorting to select the next generation of individuals. If cases where individuals have the same ranking score, the crowding distance assignment method which is based on density estimation, is used to select those individuals situated in the least crowded regions within their rank. SPEA2 calculates the strength of the individuals by considering the domination and density information, to select the new population for the next generation. The obtained Pareto front set is updated with the new population, and we use the elements in this set to calculate the hypervolume at each generation of the evolution to monitor the convergence speed of the EMO algorithms.

The EMOs were developed using the Distributed Evolutionary Algorithms in Python (DEAP) (Fortin et al., 2012) library.

4.3.4. Performance metrics for evolutionary multiobjective optimisation

Multiobjective evolutionary algorithms have three important goals (Zitzler et al., 2000): i) to minimise the distance between the obtained non-dominated solution set and the true Pareto-optimal set; ii) to obtain a good, in most cases uniform distribution of solutions; and iii) to maximise the extent of the obtained non-dominated solutions. Therefore, a wide variety of performance metrics (Riquelme et al., 2015) have been proposed to measure and compare the performance of EMO algorithms. In this study, we selected three of the most frequently used performance metrics for EMO: hypervolume, spread, and inverted generational distance (IGD), to compare the performance of EMO algorithms.

The **hypervolume** (Zitzler and Thiele, 1998), also known as the *S-metric* or *Lebesgue measure* is used for comparing convergence and diversity of a Pareto front. It measures the size of the volume of the region between the estimated Pareto-optimal front, P and, a reference point r. We followed common practice by calculating r as a point defined by taking the worst known values for each of the objectives and shifted it slightly towards some unattainable values to ensure it is placed in a manner where it will be dominated by all the other values. The hypervolume is defined as follows:

$$\mathrm{HV(P,r)} = \mathcal{L}\left(\bigcup_{i=1}^{|P|}[\mathrm{r,i}]\right) \tag{7}$$

It is the union of Lebesgue measure L for all points in P with respect to the reference point r. A Pareto front with a larger hypervolume is considered to indicate a better-performing EMO algorithm. The **spread** metric $\triangle$ (Deb et al., 2002), also known as the diversity metric, measures the diversity of the solution by calculating Euclidean distance, $\mathrm{d_i}$, between consecutive solution points. $\mathrm{d_f}$ and $\mathrm{d_l}$ are the Euclidean distances between the extreme solutions and the boundary solution of the Pareto front (Deb et al., 2002). Assuming there are N solutions in the obtained Pareto front, i = 1,i = 2,..,N-1, $\bar{\mathrm{d}}$ is the average of all Euclidean distances $\mathrm{d_i}$. A smaller value of $\triangle$ is desired, indicating a better spread of solutions. $\triangle$ is defined as follows:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1}|d_i - \bar{d}|}{d_f + d_l + (N-1)\bar{d}} \tag{8}$$

The **inverted generational distance (IGD)** (Coello and Sierra, 2004) is another widely used metric to measure both convergence and diversity. IGD is defined as:

$$IGD = \frac{\sqrt{\sum_{i=1}^{P^*} d(i,P)}}{|P^*|} \quad (9)$$

where P* denote the number of solutions in the optimal Pareto front, and d_i is the Euclidean distance between solution i and the nearest member in the obtained Pareto front, P (Van Veldhuizen and Lamont, 2000). A value, near IGD = 0 indicates better coverage of Pareto front and near true Pareto front. Due to size of the problem it was not possible to calculate the true Pareto front, and instead, a reference true Pareto front was computed for each problem by aggregating the obtained non-dominated solutions from all runs to obtain a single front. These values were used to estimate the extreme values for the spread metric, and the reference Pareto front for the IGD metric.

To compare the performance of the EMO algorithms, the Kruskal-Wallis test was employed for data that do not conform to a normal distribution, while a one-way ANOVA test was utilised for data that adhered to a normal distribution. The confidence interval for all experiments is 95%.

4.4. Random generation of combinatorial attacks

To determine the effectiveness of using EMO algorithms for attack generation a set of experiments were conducted. These involved generating random combinatorial attacks and comparing the performance of random generation with those achieved through the EMO approach. As discussed earlier, there are a total of 25 sensors and actuators that can be targeted, with four potential types of attacks: Denial of Service (DoS), minimal integrity ($integrity_{min}$), maximal integrity ($integrity_{max}$), and replay ($replay$). For the comparison study, we decided to simplify the scope of our investigation by focusing on shutdown attacks.

10 sets of 50,000 randomly generated attack strategies were generated, limiting the number of targets in each attack to a maximum of 7 out of the possible 25. For each set, a new seed was used for the random number generator of the TE model to ensure randomness in the plant simulation process. Attack types and targets (sensors and actuators to attack) were randomly selected. Attacks were started at hour 2, and they were left to run until the plant's simulation period, covering the remaining 70 hours.

To compare the performance of the EMO against the randomly generated combinatorial attacks, 10 sets of experiments were carried out using SPEA2 algorithm and the genetic operators in Table 1 with a cross-over probability of 0.9, mutation probability of 0.05, and the independent probability of each gene to be mutated at a rate of 0.05. All experiments started from a random initial population of 100 individuals and ran for 500 generations.

5. Experimental results and analysis

In this section, we discuss the experimental results and analysis. To evaluate the effectiveness of the proposed EMO approach, we first report the performance of the single-variable attacks that involved attacks on a single variable (a sensor or an actuator). The performance of the ML-based detection methods are reported in Subsection 5.2. Subsection 5.3 compares the performance of randomly generated attacks to the EMO approach. The remaining subsections report the results related to the EMO-generated attacks.

5.1. Single variable attacks

Table 2 and Table 3 shows the impact of each attack in terms of the operating cost of the plant and safety (fastest shutdown time). The *Variable Name and Range* column describes the name of the sensor or actuator, and the observed minimum and maximum values, which are used in the development of the integrity attacks. The *Shutdowns* column denotes the total number of times the plant shut down as a consequence of the attack out of 500 runs. The *Shutdown Range* column indicates the time it took the plant to shut down after the attack was started. Our experiments show that the fastest attacks that could shut down the plant are an $integrity_{min}$ attack on A and C feed flow (XMEAS 4), requiring an attack to last for a duration of 0.52-0.65 hours (31.2-39 minutes), and an $integrity_{max}$ attack on reactor temperature (XMEAS 9), resulting in a shut down at between 0.59-0.63 hours (35.4-37.8 minutes). In these cases, the controller receives altered values from XMEAS 4 and XMEAS 9, which are lower for XMEAS 4 and higher for XMEAS 9 than the normal ones. Subsequently, the controller uses these manipulated values to calculate the control signals, and transmits them to the actuators. This behaviour leads to a significant increase in the reactor pressure, causing the plant to shut down.

As reported in Table 2, the experiments carried out showed the following single attacks against process measurements (XMEAS) signals increased the operating cost of the plant significantly: i) $integrity_{max}$ attack on reactor pressure (XMEAS 7) increased the operating cost to $24,507; ii) $integrity_{max}$ attack on the sensor measuring component C in purge (XMEAS 31) increased the operating cost to $20,515; iii) DoS attack on reactor pressure (XMEAS 7) increased the operating cost to $24,299; iv) replay attack on recycle flow (XMEAS 5) increased the operating cost to $22,429; and v) DoS attack on C in purge (XMEAS 31) increased the operating cost to $17,205.

Table 3 shows the impact of attacking the manipulated variables issued by the controller to actuators (XMVs). The attack that caused the fastest damage is the $integrity_{min}$ attack on the condenser cooling water flow (XMV 11), resulting in a shutdown time of 0.64 hours (38.4 minutes) due to low separator liquid level. Carrying out a $integrity_{max}$ attack on reactor cooling water flow (XMV 10) is able to shut down the plant in 1.65 hours (99 minutes) due to high reactor pressure. A DoS attack on A and C feed flow (XMV 4) have the potential to increase the operating cost to $14,972. The integrity attacks on A and C feed flow (XMV 4), purge valve (XMV 6) and reactor cooling water flow valve (XMV 10) have also the potential to increase the operating cost; however, the impact ($9,064-11,596) is smaller compared to attacks on XMEAS, as shown in Table 3.

5.2. Detection methods

The performance of the detection algorithms on test data, unseen cases of integrity attacks on both XMEAS and XMV, are presented in Table 4 and the precision-recall curve is illustrated in Fig. 3.

Systems like the TE plant are prone to natural noise due to the behaviour of the physical components of the systems, such as actuators and sensors degrading over time, components of the plant wearing, or other forms of natural noise in the environment. The attack detection system should be robust against natural noise, and have the ability to distinguish between typical plant disturbances and attack conditions. Fig. 4 shows random forest classifying data points for a normal execution of the plant, under no attack, for 72 hours. False positives are the lines pointing at 1.

Declaring that an attack is taking place at present requires the detection to be robust to false positives. To cater for this, we used a sliding window of size 100 to declare that an attack is present only if the percentage of anomalous data points in a window exceeds a threshold to ensure false positives do not overwhelm the operator.

We executed the TE model 1000 times under normal conditions, without any attacks, using a different random seed for each replicate to achieve randomness in all the 1000 runs of the model. For each detection method, we calculated the maximum false positive percentage encountered during each run. The obtained results are presented in Fig. 5. Based on these results, the plant operator will need to define a

Table 2
Impact of single variable attacks against XMEAS signals (500 runs for each attack against each variable).

Variable Number	Variable Name and Range	Attack	Max Cost	Mean Cost	Shutdowns	Shutdown Range(hrs)
XMEAS 1	A-Feed (stream 1)	Max	8449	8407	0	-
	0.25-0.27 kscmh	Min	8364	8260	0	-
		DoS	8447	8331	0	-
		Replay	8429	8316	0	-
XMEAS 2	D Feed (stream 2)	Max	1158	1152	500	3.43-3.48
	3579.20-3744.46 kgh^{-1}	Min	915	902	500	4.31-4.4
		DoS	3149	1761	500	6.9-21.86
		Replay	3897	2765	500	13.64-30.42
XMEAS 3	E Feed (stream 3)	Max	739	730	500	2.66-2.72
	4339.06-4536.27 kgh^{-1}	Min	1320	1312	500	4.17-4.22
		DoS	2480	1494	500	3.57-18.16
		Replay	3350	2330	500	11.65-22.9
XMEAS 4	A and C Feed (stream 4)	Max	384	378	500	1.25-1.34
	8.98 9.48 9.24 kscmh	Min	392	361	500	**0.52-0.65**
		DoS	1318	743	500	1.38-6.48
		Replay	1227	825	500	2.32-5.99
XMEAS 5	Recycle flow (stream 8)	Max	2099	2040	500	8.15-8.42
	31.32-33.13 kscmch	Min	3456	3415	500	10.58-10.78
		DoS	21942	10675	322	10.4-69.7
		Replay	**22429**	9169	3	64.67-64.67
XMEAS 7	Reactor pressure	Max	**24507**	24468	0	-
	2793.54-2806.12 kPa	Min	671	650	500	8.37-8.78
		DoS	**24299**	10430	254	9.01-60.3
		Replay	23889	10894	233	9.73-66.15
XMEAS 8	Reactor level	Max	381	375	500	2.81-2.87
	62.77-67.24%	Min	1017	1008	500	2.83-2.89
		DoS	7435	1989	494	3.77-35.79
		Replay	**11302**	5058	418	15.67-67.34
XMEAS 9	Reactor temperature	Max	364	356	500	**0.59-0.63**
	122.85-122.95 °C	Min	377	366	500	1.23-1.28
		DoS	10464	1554	489	0.92-61.64
		Replay	**10774**	3681	467	2.35-67.04
XMEAS 10	Purge rate (stream 9)	Max	8228	8195	0	-
	0.1545-0.2689 kscmch	Min	8246	8218	0	-
		DoS	8235	8201	0	-
		Replay	8236	8201	0	-
XMEAS 11	Product separator temperature	Max	**10427**	10364	0	-
	91.46-92.12 °C	Min	10444	10358	0	-
		DoS	10386	10270	0	-
		Replay	**10393**	10264	0	-
XMEAS 12	Product separator level	Max	896	888	500	5.85-5.95
	45.28-54.74 mol%	Min	1256	1245	500	8.57-8.68
		DoS	8210	3816	463	8.38-66.43
		Replay	8217	7802	143	39.66-69.96
XMEAS 14	Product separator underflow	Max	1370	1357	500	9.09-9.79
	51.64-55.89%	Min	1449	1385	500	9.71-10.41
		DoS	4038	2274	500	11.07-32.98
		Replay	4431	3004	500	19.17-36.51
XMEAS 15	Stripper level	Max	1756	1740	500	13.48-13.71
	45.29-54.57%	Min	1804	1793	500	13.31-13.54
		DoS	8234	5368	413	19.77-69.61
		Replay	8234	8181	16	57.03-68.13
XMEAS 17	Stripper underflow (stream 11)	Max	312	305	500	**1.02-1.03**
	22.37-23.41 $m^3\ h^{-1}$	Min	387	379	500	1.01-1.02
		DoS	4512	2298	500	6.68-37.75
		Replay	5716	4312	500	28.45-48.24
XMEAS 31	C in Purge	Max	**20515**	20438	500	65.65-66.1
	11.87-14.22 mol%	Min	13493	13455	500	50.02-50.4
		DoS	**17205**	9841	0	-
		Replay	10951	8818	0	-
XMEAS 40	G in product	Max	8312	8298	0	-
	51.64-55.89 mol%	Min	8117	8106	0	-
		DoS	8267	8208	0	-
		Replay	8268	8212	0	-

Operating cost without attacks: Minimum: 8195, Maximum: 8218, Mean: 8208.

Table 3
Impact of single variable attacks against XMV signals (500 runs for each attack against each variable).

Variable Number	Variable Name and Range	Attack	Max Cost	Mean Cost	Shutdowns	Shutdown Range(hrs)
XMV 1	D feed flow (stream 2)	Max	8209	8198	0	-
	62.89-63.12 kgh^{-}l	Min	8226	8217	0	-
		DoS	8216	8204	0	-
		Replay	9347	8223	0	-
XMV 2	E Feed (stream 3)	Max	8234	8221	0	-
	52.99-53.24 kgh^{-}l	Min	8204	8194	0	-
		DoS	8215	8204	0	-
		Replay	8216	8203	0	-
XMV 3	A Feed (stream 1)	Max	8352	8342	0	-
	25.12-27.045 kscmh	Min	8259	8248	0	-
		DoS	8277	8216	0	-
		Replay	8247	8210	0	-
XMV 4	A and C Feed (stream 4)	Max	**10101**	10085	0	-
	59.93-61.32	Min	9339	9290	500	38.49-39.08
		DoS	**14972**	8433	2	68.09-68.09
		Replay	8873	8248	0	-
XMV 6	Purge valve (stream 9)	Max	**9552**	9542	0	-
	19.39-32.76%	Min	7223	7215	0	-
		DoS	**9064**	8221	0	-
		Replay	8876	8234	0	-
XMV 7	Separator pot liquid flow (stream 10)	Max	2382	2313	500	17.65-18.92
	37.21-37.46 m^3 h^{-}l	Min	3355	3273	500	25.83-27.35
		DoS	8217	7578	131	26.01-69.42
		Replay	8217	7975	82	44.05-68.92
XMV 8	Stripper liquid product flow (stream 11)	Max	2014	1962	500	15.18-15.99
	46.36-46.55 m^3 h^{-}l	Min	2097	2060	500	15.38-16.11
		DoS	8231	5698	366	22.32-68.72
		Replay	8235	6387	356	24.54-69.52
XMV 10	Reactor cooling water flow	Max	786	701	500	1.65-2.01
	35.46-36.33 m^3 h^{-}l	Min	**11703**	6651	489	18.88-67.49
		DoS	6093	1976	500	6.18-34.73
		Replay	5909	2283	500	6.4-34.61
XMV 11	Condenser cooling water flow	Max	325	319	500	1.59-1.6
	5.20-19.69 m^3 h^{-}l	Min	414	408	500	**0.64-0.65**
		DoS	10803	2268	477	1.61-54.62
		Replay	**11596**	7782	108	38.32-69.49

Operating cost without attacks: Minimum: 8195, Maximum: 8218, Mean: 8208.

Table 4
Performance of Detection Methods.

Detection Method	Accuracy	Precision	Recall	F1	AUC	FPR
Decision Tree	0.9632	0.951	0.8093	0.8744	0.8972	0.0078
Random Forest	0.9678	0.9557	0.835	0.8913	0.9629	0.0073
AdaBoost	0.9464	0.9709	0.6819	0.8011	0.9636	0.0038
One-Class SVM	0.9597	0.94	0.7981	0.8625	0.9619	0.0098

threshold for raising an alarm or declaring an attack is taking place. For this study, we select percentiles as the threshold for false alarm: 99% for decision tree; 99% random forest; 98% One-Class SVM; and 47% for AdaBoost. The objective of the attacker is not to raise any alarms while causing some damage by minimising the probability of detection by reducing the number of attack data points in the sliding window below the specified threshold to ensure no alarms are raised.

5.3. Comparison of random generation and EMO approach

In this section, we discuss the results obtained from experiments to compare the random generation of shutdown attacks with the EMO approach.

After removing duplicates, random generation produced a total of 442,125 unique attacks. Just over 18.5% of these attacks were able to bring the plant down in less than 1 hour. Fig. 6 illustrates the distribution of these attacks. The best attack strategy random search generated was the attack that shut down the plant in 0.158 hours (9.48 minutes) by attacking 6 sensors and actuators (XMEAS4, XMEAS8, XMEAS10, XMEAS11, XMEAS17, XMV1) using $integrity_{max}$ attack. This attack outperformed the most effective single variable attacks on single variables, bringing down the plant in 0.52 hours (31.2 minutes).

Fig. 7 shows the results obtained using the EMO approach. In total, EMO generated 35,658 unique attacks, and 86% of these attacks were under 1 hour, as indicated by the distribution skewed towards the right in Fig. 7. These attacks performed far better than those generated by random generation resulting in a high number of attacks that led to a shutdown within a timeframe of 0.138-0.156 hours. The results show the EMO approach is superior in generating attacks compared to random generation, validating our work to use the EMO approach for further attack generation. EMO approach has additionally generated better attacks than the single variable attacks shown in Table 2 and Table 3. The fastest attack the EMO generated has a shutdown time of 0.138 hours (8.28 minutes) which is significantly shorter than the

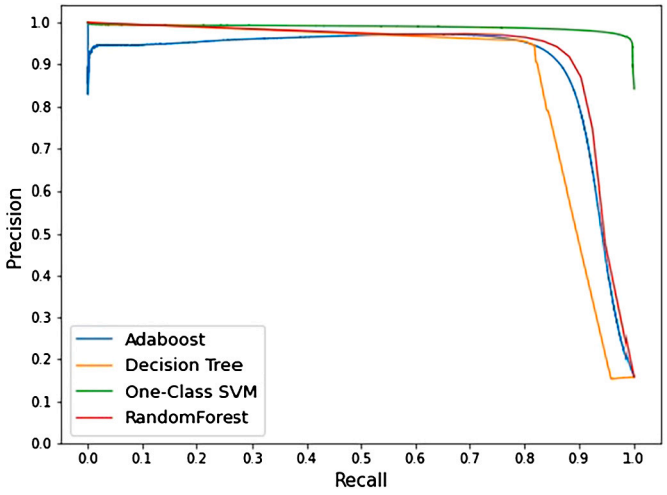


Fig. 3. Precision and Recall Curve.

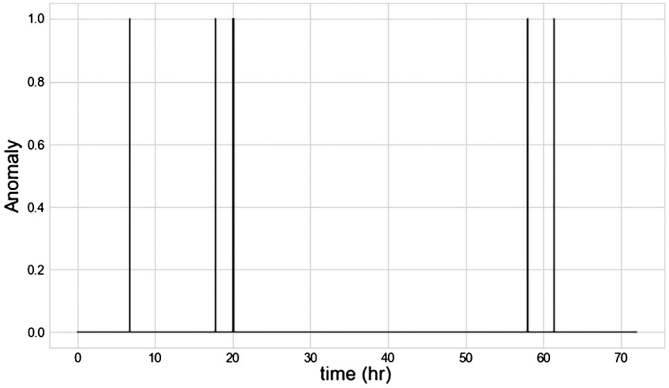


Fig. 4. Anomaly detection using random forest (normal operating conditions).

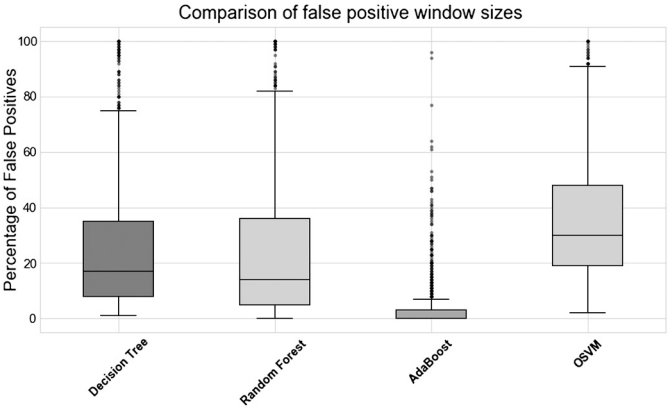


Fig. 5. Percentage of false positives for detection methods in a window size of 100 (under normal operating conditions).

fastest single variable attack, which took 0.52 hours (31.2 minutes). Similarly, the single variable attack that caused the highest increase in plant operating costs amounted to $24,507, which is much lower than the EMO-generated attacks, which escalated costs to up to $56,090. The results obtained from generating attacks using both EMO algorithms are presented in the following subsections.

5.4. Attacking the safety of the plant: shutdown attacks

In this section, we report the performance of EMO algorithms that targeted the safety of the plant by searching for attacks that shut down

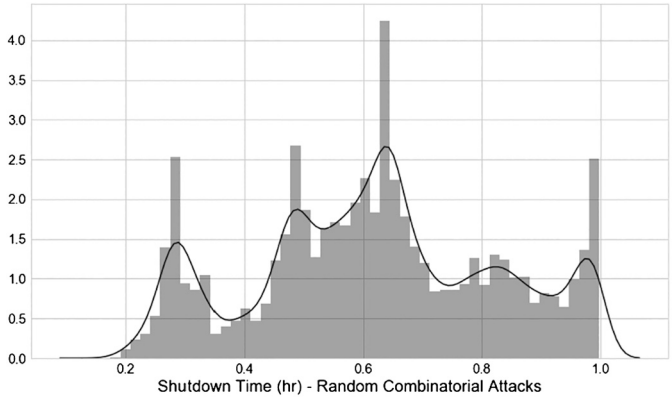


Fig. 6. Distribution of shutdown attacks generated using random generation.

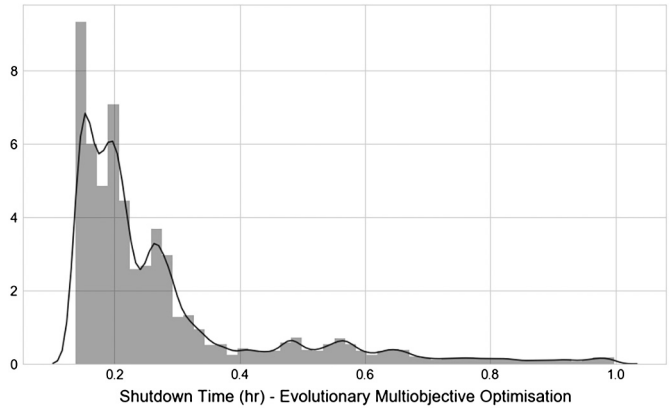


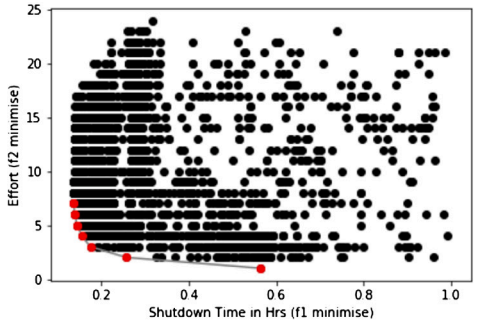
Fig. 7. Distribution of shutdown attacks generated using EMO approach.

Table 5
Performance of EMO for shutdown and operating cost attacks, averaged over all runs.

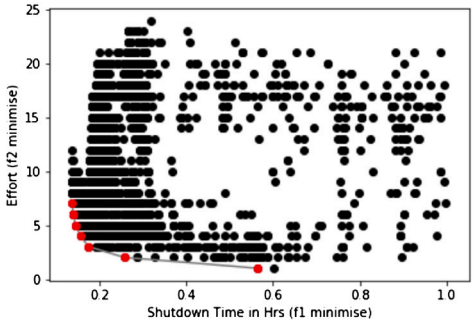
Performance Measure	NSGA-II	SPEA2	p-value
Shutdown Attacks			
Hypervolume	0.8782 (0.0316)	**0.8963** (0.0064)	**0.000486**
Spread	**0.6878** (0.2346)	0.8786 (0.1473)	0.000054
IGD	0.0598 (0.025)	**0.0368** (0.0154)	**0.000245**
Fastest Shutdown (hrs)	0.138	0.138	
Opcost Attacks			
Hypervolume	0.8235 (0.055)	**0.8877** (0.074)	**0.003**
Spread	**0.7117** (0.0853)	0.7356 (0.095)	0.234
IGD	0.1861 (0.0554)	**0.1110** (0.0494)	**0.000260**
Max. Operating Cost ($)	46,814	**56,090**	

the plant by attacking the least number of sensors and actuators. To compare the performance of NSGA-II and SPEA2 algorithms, results were collected over thirty runs for each EMO algorithm using a cross-over probability of 0.9, a mutation probability of 0.05, and the independent probability of each gene mutating at a rate of 0.05. For each of the thirty runs of evolution, a new seed was used to produce a different initial random population of size 100. The same sets of seeds were used for NSGA-II and SPEA2 to ensure both algorithms started with the same initial population. Similarly, the seed used for the TE Plant was kept the same for both algorithms. Each evolution was run for 500 generations. All experiments were carried out on an HPC platform facility at the University College London.

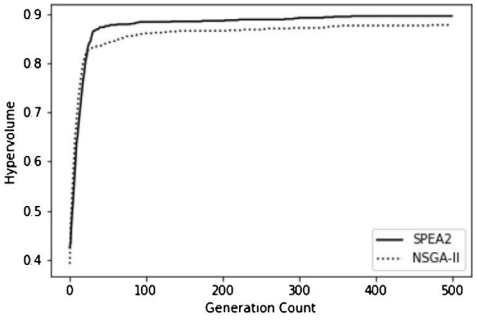
Table 5 shows the average and standard deviation of hypervolume, spread and IGD metrics. SPEA2 achieved better hypervolume and IGD with statistical confidence, yielding a better Pareto front and converging faster than NSGA-II. Meanwhile, NSGA-II obtains lower hypervolume



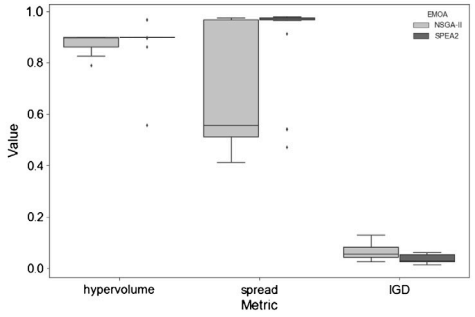
(a) Pareto front for shutdown attacks < 1 hour (NSGA-II)



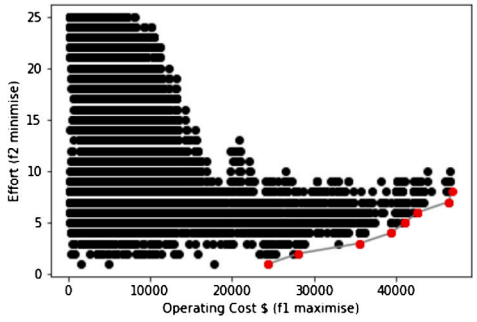
(b) Pareto front for shutdown attacks < 1 hour (SPEA2)



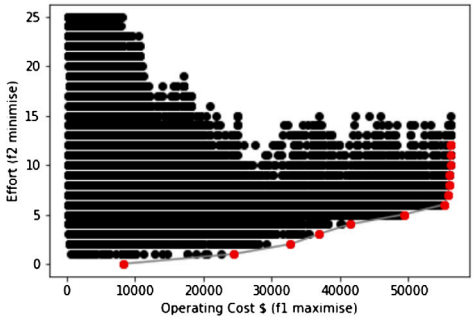
(c) Hypervolume for shutdown attacks



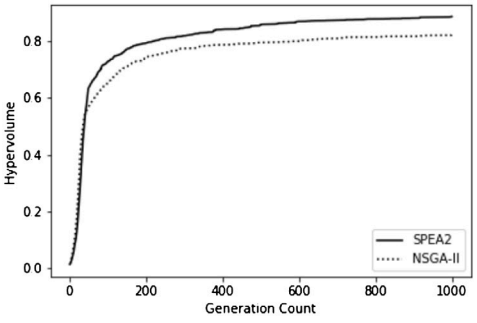
(d) Measurement metrics for shutdown attacks



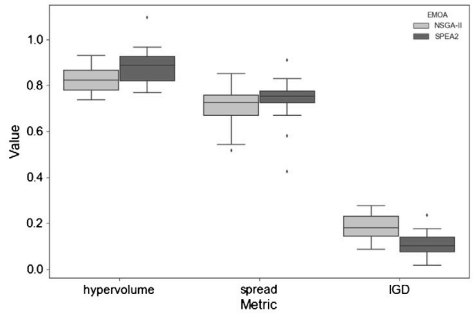
(e) Pareto front for operating cost attacks (NSGA-II)



(f) Pareto front for operating cost (SPEA2)



(g) Hypervolume for operating cost attacks



(h) Measurement metrics for operating cost attacks

Fig. 8. Results for shutdown and operating cost attacks. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

Table 6
Some elements of the Pareto front for shutdown attack generated by SPEA2.

Attack Strategy	Shut-down (hrs)	Effort
$XMEAS4_{integrity_{min}}$,$XMEAS5_{integrityMin}$,$XMEAS8_{IntegrityMin}$,$XMEAS11_{IntegrityMin}$, $XMEAS17_{IntegrityMin}$,$XMEAS31_{IntegrityMin}$, $XMV6_{Replay}$,	0.138	7
$XMEAS4_{IntegrityMin}$,$XMEAS8_{IntegrityMin}$,$XMEAS11_{IntegrityMin}$,$XMEAS17_{IntegrityMin}$, $XMEAS31_{IntegrityMin}$, $XMV_{6_{Replay}}$	0.140	6
$XMEAS4_{IntegrityMin}$,$XMEAS8_{IntegrityMin}$,$XMEAS11_{IntegrityMin}$,$XMEAS17_{IntegrityMin}$, $XMEAS31_{IntegrityMin}$	0.1460	5
$XMEAS4_{IntegrityMin}$,$XMEAS8_{IntegrityMin}$,$XMEAS11_{IntegrityMin}$, $XMEAS17_{IntegrityMin}$	0.1560	4
$XMEAS4_{IntegrityMin}$,$XMEAS8_{IntegrityMin}$,$XMEAS11_{IntegrityMin}$	0.1760	3
$XMEAS8_{IntegrityMin}$,$XMEAS11_{IntegrityMin}$	0.2579	2
$XMEAS4_{IntegrityMin}$	0.5640	1
Do Not Attack	0	0

Table 7
Some elements of the Pareto front for operating cost attacks generated by SPEA2.

Attack Strategy	Opcost ($)	Effort
$XMEAS2_{DoS(2,70)}$,$XMEAS7_{IntegrityMax(2,70)}$,$XMEAS8_{DoS(10,20)}$,$XMEAS11_{DOS(30,42)}$, $XMEAS12_{Replay(50,12)}$,$XMEAS15_{IntegrityMax(50,12)}$,$XMEAS31_{IntegrityMin(30,42)}$, $XMV1_{DOS(10,62)}$,$XMV2_{IntegrityMin(2,70)}$,$XMV3_{DOS(2,70)}$,$XMV4_{DOS(50,12)}$,$XMV11_{DOS(50,12)}$	56,090	12
$XMEAS2_{DoS(2,70)}$,$XMEAS7_{IntegrityMax(2,70)}$, $XMEAS31_{IntegrityMin(30,42)}$,$XMV1_{DoS(10,62)}$, $XMV2_{IntegrityMin(2,70)}$,$XMV3_{DoS(2,70)}$	55,275	6
$XMEAS2_{DoS(2,70)}$,$XMEAS7_{IntegrityMax(2,70)}$, $XMEAS31_{IntegrityMin(30,42)}$, $XMV1_{DoS(10,62)}$, $XMV3_{IntegrityMin(2,70)}$	49,449	5
$XMEAS2_{DoS(2,70)}$,$XMEAS7_{IntegrityMax(2,70)}$,$XMV1_{DoS(10,62)}$,$XMV3_{IntegrityMin(2,70)}$	41,430	4
$XMEAS7_{IntegrityMax(2,70)}$,$XMEAS31_{DoS(10,62)}$,$XMV3_{IntegrityMin(2,70)}$	36,866	3
$XMEAS7_{IntegrityMax(2,70)}$,$XMEAS31_{IntegrityMin(30,42)}$	32,761	2
$XMEAS7_{IntegrityMax(2,70)}$	24,479	1
Do Not Attack	8,210	0

and IGD than SPEA2 despite producing a significantly better spread. The solutions in the obtained Pareto front were not widely spread across the front, only a small number of signals needed to be altered to cause the plant to shut-down. SPEA2 was able search this area better, and converged to a better Pareto front faster than NSGA-II.

One of the best runs obtained from shutdown attacks using NSGA-II and SPEA2 is shown in Fig. 8a and Fig. 8b. The elements of the final Pareto front obtained at the end of the evolution are plotted in red dots, and some of the members of the Pareto set are shown in Table 6. For this particular run, SPEA2 outperformed NSGA-II in that it found several different attacks with equal fitness functions and effort values in the Pareto set with better Pareto front cardinality; in this case, a total of 12 elements, against the 9 found by NSGA-II. In the plot of the Pareto front, some of the points on the diagram refer to multiple attacks with equal fitness; for example, there were 3 attacks using 6 effort and a shutdown time of 0.140 hours; 2 attacks with 4 effort and shutdown time of 0.1560 hours.

Fig. 8c shows the hypervolume for each of the 500 generations, averaged over all 30 runs. At the end of each generation, the hypervolume was computed according to the Pareto front achieved at that generation to compare the speed of the convergence. For convenience, plotted hypervolume results are normalised to the interval [0-1] according to the best hypervolume value possible, estimated based on the maximum measurement obtained. SPEA2 converges faster to a better Pareto front whereas NSGA-II requires more time to reach a slightly worse Pareto set. This is supported by IGD metric, as shown on the boxplot in Fig. 8d SPEA2 scores a better IGD score.

Results obtained show that combinatorial attacks generated using the EMO approach performed better than single and combinatorial attacks generated randomly against the XMEAS and XMV variables. The single attacks were able to shut down the plant in 0.52 hours, whereas, EMO found a range of attacks that could bring down the plant much faster, in 0.138 hours. Random attacks at the very best produced an attack that could bring down the plant in 0.158 hours.

5.5. Causing economic loss: operating cost attacks

In this section, we report the performance of EMO algorithms that targeted the operating cost of the plant to cause economic loss by attacking the least number of sensor and actuator signals. Due to the slowness of the TE model, only a small set of attack start times (2, 10, 20, 30, 50 hours) and attack duration (10, 12, 20, 42, 50, 52, 62, 70 hours) were used to generate attacks. As before, attack types were DoS, replay, $integrity_{max}$ and $integrity_{min}$. Individuals were represented as chromosomes encoded as a list of 25 integers (genes), each position denoting a sensor or an actuator. In total, each gene could have a value between 0-37 (0 representing *not to attack*, the remaining 36 representing 9 different attacks per attack type with different start times and attack duration). This is a search space of size 37^{25}. Twenty runs were carried out for each EMO algorithm to analyse the impact of attacks on the operating cost of the plant. As before, attacks were generated using the same genetic operators, shown in Table 1, with a cross-over probability of 0.85, mutation probability of 0.10, and probability of mutating a gene of 0.05. Each evolution started with a random population of 400 individuals and ran for 1000 generations.

As with shutdown attacks, SPEA2 performed better than NSGA-II both in the quality of obtained Pareto front set, and the time it took to converge. As indicated in Table 5, the average over all the runs showed that SPEA2 has a hypervolume average of 0.8877, against NSGA-II scoring a hypervolume of 0.8235. SPEA2 produced significantly higher hypervolume and IGD, but no significant difference was found between the two algorithms for spread. Fig. 8g shows the comparison of hypervolume between NSGA-II and SPEA2, averaged over all runs. These results show that SPEA2, as before, is able to search the space faster and produce a better Pareto front with a higher cardinality. SPEA2 (Table 7) increased the cost of operating the plant from an average of $8,208 to $56,090, whereas NSGA-II increased the operating cost to $46,814. Table 7 shows some of the elements of the obtained Pareto front set for SPEA2. The numbers in the bracket denote the start time and duration of the attack, for example, $XMEAS8_{DoS(10,20)}$ means a DoS attack was carried out against XMEAS 8 starting at hour 10, for a duration of 20

hours. Overall, these results show that EMO approach can successfully be used to generate attacks that could cause economic loss, by identifying the components that increase the operating cost of the plant.

5.6. Generating attacks against detection methods

As in previous instances, individuals generated against detection methods are represented as a list, consisting of 25 positions, each position denoting a sensor or an actuator. One key obstacle we had with our experiments was time, the execution of the TE model in MATLAB could take up to several minutes. This performance issue influenced the way we encoded our individuals, the size of the population, and the number of generations. To make the problem computationally tractable, we limited the types of genes to a pool of 140 in which half denoted DoS attacks and the remaining half denoted replay attacks. Genes were represented as integers, each number denoting the start time of the attacks (between hour 2-70). The duration of the attacks against detection methods were kept constant for all attacks, 2 hours. This is a combinatorial search problem of size 140^{25}. Results were collected over 10 runs for each EMO algorithm against each detection method (AdaBoost, decision tree, random forest and one-class SVM) using a cross-over probability of 0.8, mutation probability of 0.15, and probability of mutating a gene at a rate of 0.08. Each evolution started from a random population of 200 individuals and ran for 1000 generations. Attacks that caused the plant to shut down were penalised, with a score -1, as our aim was to generate attacks that kept the plant running while causing some damage and evading detection.

Despite using a limited number of attack parameters (start time, duration), and using a low population and generation size, the obtained result captures some of the weaknesses of the detection methods.

Fig. 9 shows the results obtained. Fig. 9a, Fig. 9c, Fig. 9e and Fig. 9g show one of the best Pareto fronts obtained against four classifiers. As expected, EMO algorithms were able to exploit the lower detection capability of AdaBoost, and cause more damage ($406-6) while keeping the attack detection (alarm) probability at a lower rate. EMO algorithms performed less damaging attacks against the decision tree ($190-10), random forest ($77-10), and one-class SVM ($21-13). As before, hypervolume was computed at each generation to analyse the convergence. Fig. 9b, 9d, 9f, and 9h shows a steady increase of hypervolume suggesting that more generations will yield a better search and convergence. Fig. 10 shows the hypervolume, spread and IGD boxplots for four classifiers. As indicated in Table 8 statistical test shows there was no significant difference between NSGA-II and SPEA2 against decision tree, random forest and one-class SVM. However, for AdaBoost, NSGA-II did significantly better, both in terms of obtaining a better Pareto front (as indicated by hypervolume and IGD) metrics and converged much faster as indicated by hypervolume plot (Fig. 9b). Although, this time SPEA2 had a better diversity compared to NSGA-II, it did not yield a better Pareto front.

As the damage increases, denoted by the increased cost of operating the plant, the likelihood of detection also increases. EMO algorithms failed to evolve highly damaging attacks against the one-class SVM as it was able to detect DoS and replay attacks better than other detection methods. However, we were able to cause some economic damage with low detection probabilities for other detection methods.

5.6.1. Seeding initial population, and generating attacks using 3-objective optimisation

In this section, we report some preliminary experiments that require further investigation, but show promising results. Generating attacks against a strong classifier can be a very time consuming task; this is especially true for cases like our plant model, for which the evaluation of individuals (fitness function) is slow. The slowness of our fitness function also influenced the attack parameters, the size of the population, and the number of generations. To test if we could generate better attacks against decision tree and random forest classifiers, we started some experiments using a seeded initial population that included some good individuals that were obtained previously from experiments carried out against the decision tree classifier (i.e., Fig. 9c). Seeding is a common practice in single-objective evolutionary algorithms where prior knowledge obtained from previous experiments or expert knowledge is included in the initial population as a good initial estimate, however the advantages and disadvantages of employing seeding in EMO algorithms, particularly for solving real-world combinatorial optimisation problems require further studies (Friedrich and Wagner, 2015).

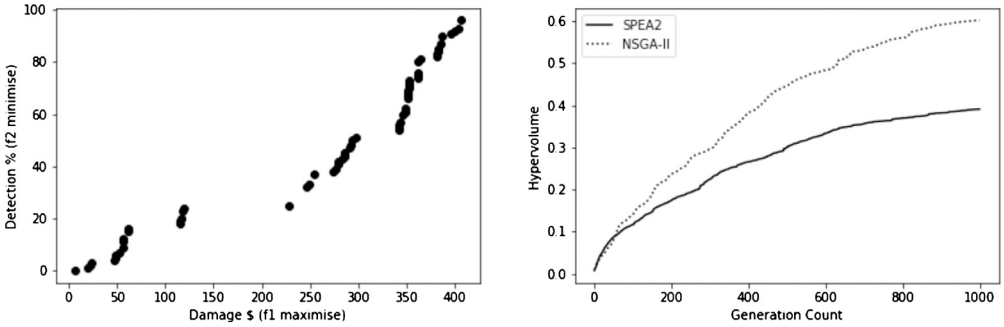
Table 8
Performance of EMO generated attacks against detection, averaged over all runs.

Performance Measure	NSGA-II	SPEA2	p-value
AdaBoost			
Hypervolume	**0.6020** (0.2257)	0.3910 (0.2525)	**0.032663**
Spread	0.7952 (0.0910)	**0.7165** (0.0841)	**0.000054**
IGD	**0.2273** (0.1024)	0.3345 (0.1203)	**0.0376**
Damage Range ($)	15-490.63	12.5-424.99	
Decision Tree			
Hypervolume	0.1442 (0.0464)	**0.1452** (0.0416)	0.8798
Spread	0.6825 (0.0564)	**0.6450** (0.0441)	0.1340
IGD	**0.1505** (0.0506)	0.1580 (0.0382)	0.7284
Damage Range ($)	9-190.63	13-182.26	
Random Forest			
Hypervolume	0.1170 (0.0290)	**0.1240** (0.0304)	0.2895
Spread	**0.5846** (0.0384)	0.5943 (0.0361)	0.5877
IGD	**0.1375 (0.0110)**	0.1382 (0.01328)	0.7622
Damage Range ($)	4-76.92	1.56-56.10	
One-Class SVM			
Hypervolume	0.0785 (0.0222)	**0.0788** (0.0225)	0.9768
Spread	**0.5887** (0.0363)	0.5909 (0.03789)	0.8205
IGD	**0.1108** (0.0389)	0.1181 (0.04144)	0.7070
Damage Range ($)	1.5-21.28	2.0-20.75	

A comprehensive study is left as future work, and here we report some initial results. We carried out an experiment where we seeded the initial population with 10 of the best individuals obtained from the previous runs of decision tree experiments, and started the EMO from this modified population against the decision tree and random forest classifiers. Fig. 11a shows the performance of decision tree after seeding the initial population (compare with no seeding in Fig. 9c), and Fig. 11b shows the performance of random forest after seeding the initial population (compare with no seeding in Fig. 9e), showing attacks with higher damage and low detection probability. These results indicate seeding could significantly reduce the duration of experiments, and more rapidly identify those attacks that are likely to evade detection (raise alarms) and, at the same time cause some economic loss. However, further experiments are necessary to understand the full benefits and weaknesses of the variety of strategies for this approach (e.g. such as the number of seeds used), as seeding can reduce the diversity of the population, and prevent the EMO from exploring other regions in the attack space.

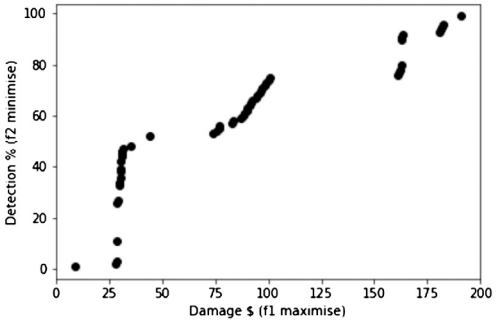
One of the weaknesses of the two objective optimisations against the detection methods was that the effort was not optimised, and we were not able to tell if attacks could be generated using less effort. To investigate if attackers could cause the same damage using less effort, we carried out a 3-objective optimisation (maximise the operating cost of the plant, minimise detection, minimise effort). Due to computational constraints, only one detection method, AdaBoost was chosen for further investigation. Table 9 shows the performance of the NSGA-II and SPEA2 averaged over two runs, for 800 generations.

Both runs required more time to converge, but NSGA-II appears to search a wider region using 3-objectives compared to SPEA2. NSGA-II showed a better spread of attacks over the search space, and a better value for hypervolume. The best attacks are those with lower detection probability situated in the lower regions of the detection causing damages of less than $200. As shown in Fig. 12, NSGA-II finds attacks with

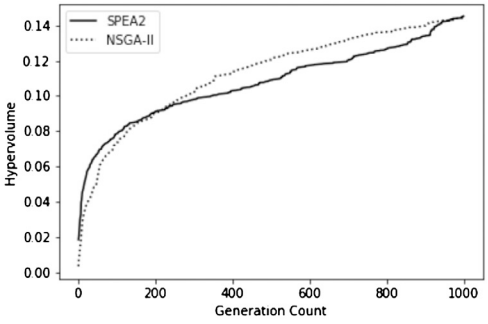


(a) Pareto front for AdaBoost

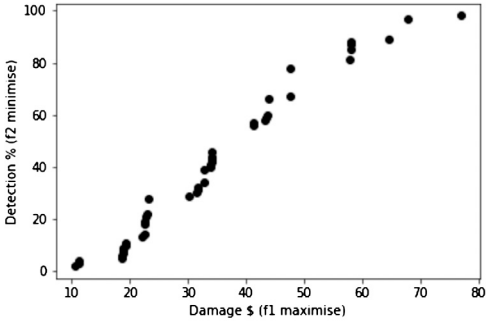
(b) Hypervolume for AdaBoost



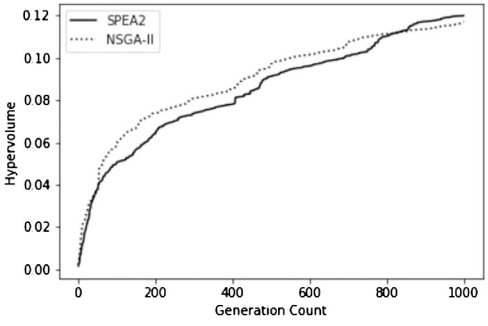
(c) Pareto front for Decision Tree



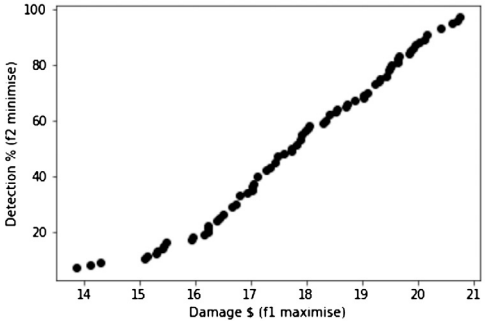
(d) Hypervolume for Decision Tree



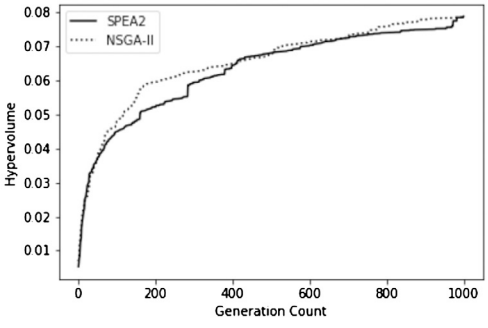
(e) Pareto front for Random Forest



(f) Hypervolume for Random Forest

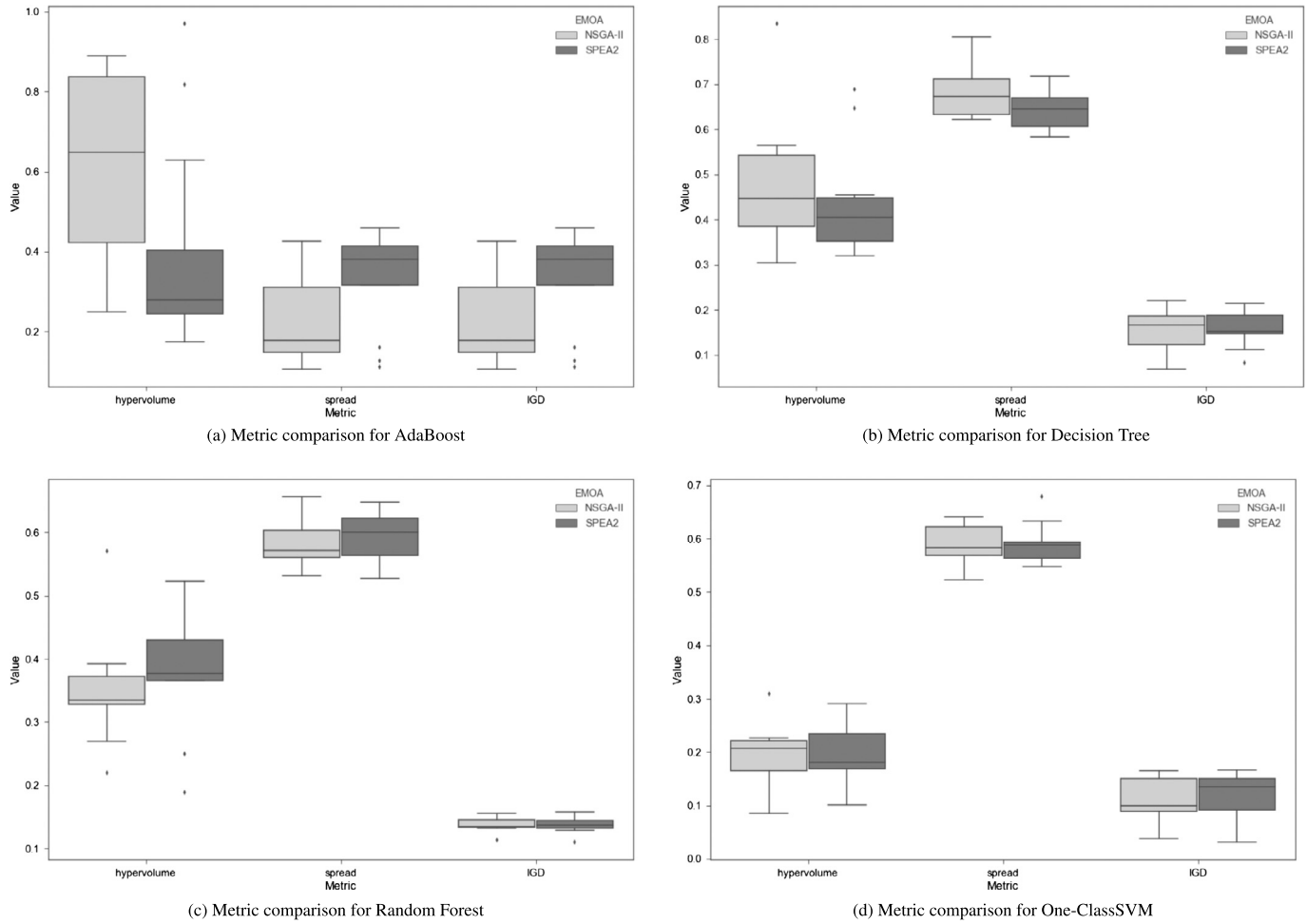


(g) Pareto front for One-Class SVM



(h) Hypervolume for One-Class SVM

Fig. 9. Pareto front and hypervolume for attacks generated against the detection methods, averaged over all runs.



(a) Metric comparison for AdaBoost

(b) Metric comparison for Decision Tree

(c) Metric comparison for Random Forest

(d) Metric comparison for One-ClassSVM

Fig. 10. Performance of detection methods, averaged over all runs.

Table 9
Performance metrics for 3-objective attacks against AdaBoost classifier.

	NSGA-II	SPEA2
Damage Range ($)	0-1633	0-1114
Hypervolume	0.7140	0.5423

higher damage (i.e. damage over $1500), but, SPEA2 tends to generate more attacks in the lower regions where there is a better probability of avoiding detection.

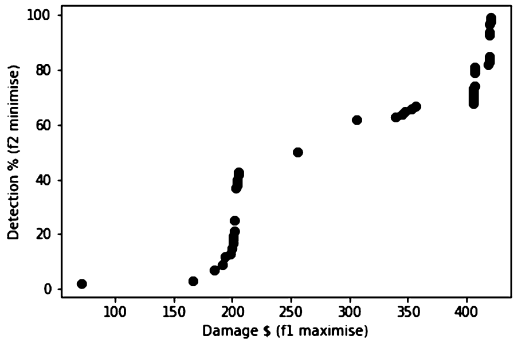
A more reliable comparison requires additional experiments, which we leave as part of future work; however, overall, both algorithms successfully identified attacks that avoid detection while inflicting a certain level of damage. Assuming the detection probability threshold is less than 5%, the highest economic damage NSGA-II could found is $266.92, attacking 12 sensors and actuators, with a detection probability of 3%, whereas SPEA2 found a slightly worse attack, a damage of $179.72 with an effort of 16 and detection probability of 4%. If the intention is to use the smallest effort and cause maximum damage, then the optimal attack strategy produced using NSGA-II is an attack that costs $179.72 using effort of 6. SPEA2 found a similar attack using an effort of 6, causing a damage of $170.69. The attacks generated using less effort were either detected or caused very little economic damage, that is ≤$50.

6. Application of results

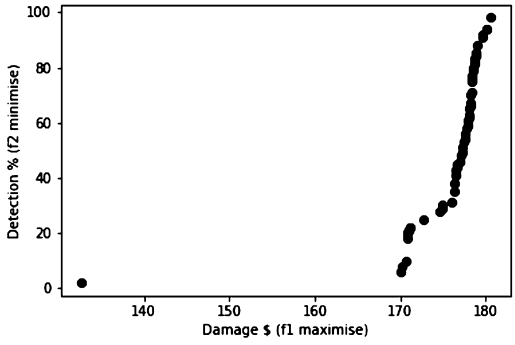
The EMO approach developed in this study can be employed to discover the vulnerabilities of cyber-physical systems that have an extensive attack surface. When targeting these systems, attackers may focus on process measurements and manipulated variables because these values can directly impact the process, and potentially cause physical harm.

The results obtained provide a detailed set of attack scenarios that plant operators can analyse to identify the most vulnerable combinations of sensors and actuators. Table 10 shows a small subset of vulnerable combinations of the sensors and actuators that could shut down the TE plant in under 17 minutes. For example, carrying out a single attack on XMEAS 8 (reactor level) was able to shut down the plant over 2.8 hours, and attacking XMEAS 11 (separator temperature) avoided shutting down the plant for all types of attacks. Results show that attacking both of these sensors simultaneously could shut down the plant in 14.8 minutes. These results also show that the plant is less resilient to attacks on process measurements (sensors), and, if an adversary wants to bring down the plant in a very short period of time such as less than 10 minutes, attacking sensors is more likely to cause this to happen than attacking manipulated variables. This is, possibly because the selected attack parameters for actuator signals (based on the observed signals), were not as effective as attacking process measurements. Similarly, DoS attacks were slower and less successful against manipulated signals. Future research will investigate this further, focusing more on the attack parameters and timing of the attacks.

Similarly, we found a wide variety of combinatorial attacks that are capable of increasing the operating cost. Most of these attacks involved carrying out an $integrity_{max}$ attack on XMEAS 7 (reactor pressure), which means sending higher values than expected. The likely reason for this is that upon receiving a high value, the controller attempts to



(a) Attack generated against Decision Tree



(b) Attack generated against Random Forest

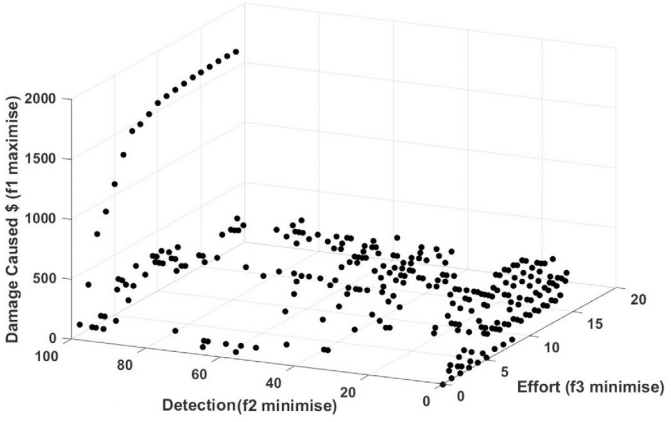
Fig. 11. Seeding population with knowledge gained from prior experiments.

Table 10
A selection of vulnerable combinations that could bring the plant down under 17 minutes.

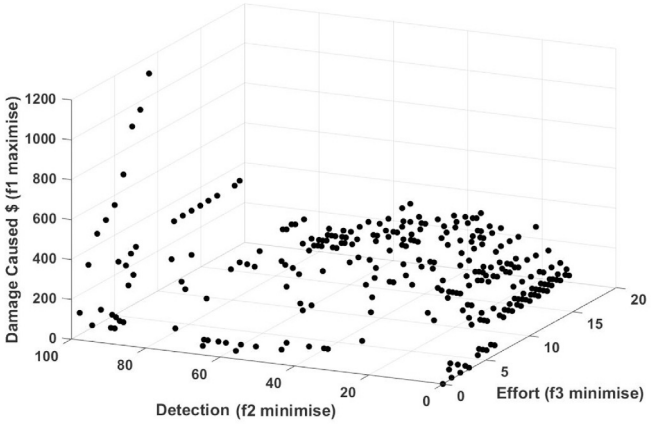
Vulnerable Combinations	SDT (min)
XMEAS4,XMEAS8,XMEAS11, XMEAS17,XMV6	8.9
XMEAS4,XMEAS8,XMEAS11, XMEAS31	9.6
XMEAS3,XMEAS4,XMEAS8, XMEAS11	10.2
XMEAS8,XMEAS9,XMEAS11	10.8
XMEAS8,XMEAS11,XMV6	11.4
XMEAS8,XMEAS11,XMV10	12.0
XMEAS5,XMEAS8,XMEAS11	13.8
XMEAS9,XMV7,XMV11	14.4
XMV7,XMV10,XMV11	14.5
XMEAS4,XMEAS7,XMEAS17	14.7
XMEAS8,XMEAS11	14.8
XMV9, XMV10	16.2
XMEAS9,XMV11	16.8

lower the pressure by opening the purge valve, as a result leading to loss of raw materials in the purge stream. This in turn increases the operating cost. Carrying out a single attack on XMEAS 7 increases the operating cost to $24,507 but, as the results indicate, more damage can be inflicted using the right combinations of sensors and actuators.

The results obtained demonstrate that the proposed EMO approach can be used to generate attacks to identify vulnerable parts of a system, and this knowledge can be leveraged to design systems that are more secure and resilient to cyberattacks. One way to achieve this is to consider the vulnerable combinations of elements when designing network segmentation. The *zone and conduit model* is a framework for network segmentation to manage security threats for industrial automation and control systems, recommended as part of the standard such as ISA/IEC 62443 (ISA, 2020). Zones are defined as a group of logical or physical assets sharing common security requirements, and conduits are the paths of communication between the zones. Leveraging the knowledge from the EMO approach, vulnerable sensor and actuator combinations that lead to plant shutdowns or increase the operating cost of the plant can be aggregated in different zones to build a more secure network.



(a) NSGA-II



(b) SPEA2

Fig. 12. Obtained Pareto front for 3-objective attacks against AdaBoost classifier.

Given the ability of EMO to generate large number of attacks that went undetected by all employed detection methods, this approach can also be used as a tool to test and develop better detection methods. However, generating attacks that evade detection, and at the same time cause some significant economic damage to a system like the TE process is a challenging task since it requires attacks to be long in duration. Our results indicate that carrying out a successful attack requires knowledge of the system to ensure that the correct combination of sensors and actuators are attacked, and to avoid attack detection or activation of the safety system. The significant attacks generated against AdaBoost, decision tree and random forest classifiers involved attacking multiple components in the system to evade detection while causing economic damage. A naïve attacker that randomly attacks multiple targets is unlikely to achieve similar damage and has a high likelihood of being detected. The focus of our future work will involve investigating and designing more complex attacks that could learn the behaviour of the plant and the detection system, and utilising this knowledge to generate more sophisticated attacks.

Overall, the obtained results show that evolutionary multiobjective optimisation can be used successfully as a tool for simulating adversarial behaviour against attack detection methods, while highlighting the inherent trade-offs among security objectives: the impact of an attack, the detection likelihood of the attack and effort required for execut-

ing the attack. Using the insight gained from such an analysis, security engineers can take measures to understand and eradicate system vulnerabilities before they are exploited by malicious actors.

7. Conclusions and future work

This paper demonstrates a novel application of evolutionary multi-objective optimisation for the security of industrial control systems, and more general cyber-physical systems. Using a simulation of a complex and realistic plant, of the sort used routinely in factories and plants, it is possible to automate the generation of combinatorial attacks to discover vulnerabilities.

The threat to such systems is both realistic and of critical importance. To the best of our knowledge, there are no methods that are both robust and efficient in identifying vulnerable combinations of components in a complex system. Our proposed approach represents a promising step towards achieving this. The security knowledge derived from the proposed EMO approach can be utilised in a number of ways. The first is in determining the criticality of security decisions such as vulnerability patching; selecting appropriate attack detection and prevention methods; and designing resilient network segments. Secondly, control engineers can use the insight gained from this work to analyse the implications of security attacks on process control, and design resilient control algorithms. The attacks generated against the detection methods show our approach can also be used as a tool to find the vulnerabilities in the detection before they are exploited by adversaries.

In this study, TE model was used as a case study to demonstrate the methodology. However, the approach is agnostic and not specific to any one system. Future work will focus on demonstrating this on other cyber-physical systems using more advanced attacks. A major obstacle to research in ICS security is the lack of readily available benchmark testbeds. We are currently working on building cyber-physical testbeds, and we hope to test our approach on a different type of industrial process with physical and network components. Finally, we plan to explore how other EMOs, particularly the newer aggregation-based (e.g. Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D) and Indicator-Based Evolutionary Algorithms (IBEA) would perform on this kind of problem.

CRediT authorship contribution statement

Nilufer Tuptuk: Conceptualization, Investigation, Methodology, Software, Visualization, Writing – review & editing. **Stephen Hailes:** Conceptualization, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Nilufer Tuptuk reports financial support was provided by Engineering and Physical Sciences Research Council.

Data availability

Data will be made available on request.

Acknowledgements

This work was conducted partly under UK EPSRC Grant No: EP/G037264/1 as part of University College London's Security Science Doctoral Training Centre, and partly under UK EPSRC Grant No: EP/N023234/1 as part of PETRAS Internet of Things Research Hub.

References

Bronfman-Nadas, R., Zincir-Heywood, N., Jacobs, J.T., 2018. An artificial arms race: could it improve mobile malware detectors? In: 2018 Network Traffic Measurement and Analysis Conference (TMA), pp. 1–8.

BSI (Federal Office for Information Security), 2014. Die lage der it-sicherheit in Deutschland 2014. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Lageberichte/Lagebericht2014.pdf. (Accessed 15 November 2023).

Cárdenas, Alvaro A., Amin, Saurabh, Lin, Zong-Syun, Huang, Yu-Lun, Huang, Chi-Yen, Sastry, Shankar, 2011. Attacks against process control systems: risk assessment, detection, and response. In: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS '11. Association for Computing Machinery, New York, NY, USA, pp. 355–366.

CBC, 2020. Cyberattack shuts down evraz it systems across North America, but company says no data compromised. Available: https://www.cbc.ca/news/canada/saskatchewan/evraz-regina-shut-down-ransomware-attack-1.5487017. (Accessed 3 November 2020).

Cerda-Flores, Sandra C., Rojas-Punzo, Arturo A., Nápoles-Rivera, Fabricio, 2022. Applications of multi-objective optimization to industrial processes: a literature review. Processes 10 (1).

Coello Coello, C.A., Reyes Sierra, M., 2004. A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm. In: Monroy, R., Arroyo-Figueroa, G., Sucar, L.E., Sossa, H. (Eds.), MICAI 2004: Advances in Artificial Intelligence. In: Lecture Notes in Computer Science, vol. 2972. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-24694-7_71.

Deb, Kalyanmoy, Agrawal, Samir, Pratap, Amrit, Meyarivan, T., 2000. A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 849–858.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Trans. Evol. Comput. 6 (2), 182–197.

Decraene, J., Chandramohan, M., Low, M.Y.H., Choo, C.S., 2010. Evolvable simulations applied to automated red teaming: a preliminary study. In: Simulation Conference (WSC), Proceedings of the 2010 Winter, pp. 1444–1455.

Dewri, Rinku, Poolsappasit, Nayot, Ray, Indrajit, Whitley, Darrell, 2007. Optimal security hardening using multi-objective optimization on attack tree models of networks. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, CCS '07. ACM, New York, NY, USA, pp. 204–213.

Di Pietro, Antonio, Foglietta, Chiara, Palmieri, Simone, Panzieri, Stefano, 2013. Assessing the impact of cyber attacks on interdependent physical systems. In: Butts, Jonathan, Shenoi, Sujeet (Eds.), Critical Infrastructure Protection VII. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 215–227.

Diaz-Gomez, Pedro A., De Sistemas, Ingenieria, Hougen, Dean F., 2005. Improved off-line intrusion detection using a genetic algorithm. In: Proceedings of the Seventh International Conference on Enterprise Information Systems.

Downs, J.J., Vogel, E.F., 1993. A plant-wide industrial process control problem. Comput. Chem. Eng. 17 (3), 245–255.

Dragos Inc., 2017. CRASHOVERRIDE Analyzing the threat to electric grid operations version 2.20170613. Available: https://dragos.com/wp-content/uploads/CrashOverride-01.pdf. (Accessed 6 April 2018).

Erba, Alessandro, Taormina, Riccardo, Galelli, Stefano, Pogliani, Marcello, Carminati, Michele, Zanero, Stefano, Tippenhauer, Nils Ole, 2019. Real-time evasion attacks with physical constraints on deep learning-based anomaly detectors in industrial control systems. CoRR. arXiv:1907.07487 [abs].

Fielder, Andrew, Panaousis, Emmanouil, Malacaria, Pasquale, Hankin, Chris, Smeraldi, Fabrizio, 2016. Decision support approaches for cyber security investment. Decis. Support Syst. 86, 13–23.

Fortin, Félix-Antoine, De Rainville, François-Michel, Gardner Gardner, Marc-André, Parizeau, Marc, Deap, Christian Gagné, 2012. DEAP: Evolutionary Algorithms Made Easy. J. Mach. Learn. Res. 13 (1), 2171–2175.

Friedrich, Tobias, Wagner, Markus, 2015. Seeding the initial population of multi-objective evolutionary algorithms: a computational study. Appl. Soft Comput. 33, 223–230.

F-Secure Labs, 2014. Havex Hunts For ICS/SCADA Systems. Available: https://archive.f-secure.com/weblog/archives/00002718.html. (Accessed 16 November 2023).

Garcia, Dennis, Erb Lugo, Anthony, Hemberg, Erik, O'Reilly, Una-May, 2017. Investigating coevolutionary archive based genetic algorithms on cyber defense networks. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '17. ACM, New York, NY, USA, pp. 1455–1462.

Genge, B., Siaterlis, C., Hohenadel, M., Genge, Béla, Siaterlis, Christos, Hohenadel, Marc, 2012. Impact of network infrastructure parameters to the effectiveness of cyber attacks against industrial control systems. Int. J. Comput. Commun. Control, 673.

Glavan, Miha, Gradišar, Dejan, Atanasijević-Kunc, Maja, Strmčnik, Stanko, Mušič, Gašper, 2013. Input variable selection for model-based production control and optimisation. Int. J. Adv. Manuf. Technol. 68 (9), 2743–2759.

Goyal, Anup, Kumar, Chetan, 2008. Ga-nids: a genetic algorithm based network intrusion detection system. Technical Report. ElectricalEngineering & Computer Science. North West University.

Hemberg, Erik, Zipkin, Joseph R., Skowyra, Richard W., Wagner, Neal, O'Reilly, Una-May, 2018. Adversarial co-evolution of attack and defense in a segmented computer network environment. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '18. Association for Computing Machinery, New York, NY, USA, pp. 1648–1655.

Hoque, Mohammad Sazzadul, Mukit, Md. Abdul, Bikas, Md. Abu Naser, 2012. An implementation of intrusion detection system using genetic algorithm. CoRR. arXiv:1204.1336 [abs].

Huang, Yu-Lun, Cárdenas, Alvaro A., Amin, Saurabh, Lin, Zong-Syun, Tsai, Hsin-Yi, Sastry, Shankar, 2009. Understanding the physical and economic consequences of attacks on control systems. Int. J. Crit. Infrastr. Protect. 2 (3), 73–83.

Huang, Ling, Joseph, Anthony D., Nelson, Blaine, Rubinstein, Benjamin I.P., Tygar, J.D., 2011. Adversarial machine learning. In: Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, AISec '11. Association for Computing Machinery, New York, NY, USA, pp. 43–58.

Huang, K., Zhou, C., Tian, Y., Yang, S., Qin, Y., 2018. Assessing the physical impact of cyberattacks on industrial cyber-physical systems. IEEE Trans. Ind. Electron. 65 (10), 8153–8162.

ISA, 2020. An Overview of ISA/IEC 62443 Standards Security of Industrial Automation and Control Systems. Available: https://gca.isa.org/hubfs/ISAGCA%20Quick%20Start%20Guide%20FINAL.pdf. (Accessed 16 November 2023).

Isakov, A., Krotofil, M., 2015. Damn Vulnerable Chemical Process - Tennessee Eastman. Available: https://github.com/satejnik/DVCP-TE.

John, David J., Smith, Robert W., Turkett, William H., Cañas, Daniel A., Fulp, Errin W., 2014. Evolutionary based moving target cyber defense. In: Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation, GECCO Comp '14. ACM, New York, NY, USA, pp. 1261–1268.

Kayacik, Hilmi Güneş, Heywood, Malcolm, Zincir-Heywood, Nur, 2006. On evolving buffer overflow attacks using genetic programming. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, GECCO '06. ACM, New York, NY, USA, pp. 1667–1674.

Kravchik, Moshe, Biggio, Battista, Shabtai, Asaf, 2021. Poisoning attacks on cyber attack detectors for industrial control systems. In: Proceedings of the 36th Annual ACM Symposium on Applied Computing, SAC '21. Association for Computing Machinery, New York, NY, USA, pp. 116–125.

Krotofil, Marina, Cárdenas, Alvaro A., 2013. Resilience of Process Control Systems to Cyber-Physical Attacks. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 166–182.

Larsson, Truls, Hestetun, Kristin, Hovland, Espen, Skogestad, Sigurd, 2001. Self-optimizing control of a large-scale plant: the Tennessee Eastman process. Ind. Eng. Chem. Res. 40 (22), 4889–4901.

Li, Wei, 2004. Using genetic algorithm for network intrusion detection. In: Proceedings of the United States Department of Energy Cyber Security Group 2004 Training Conference, pp. 24–27.

Liu, Hui, Li, Ye, Duan, Zhu, Chen, Chao, 2020. A review on multi-objective optimization framework in wind energy forecasting techniques and applications. Energy Convers. Manag. 224, 113324.

Mrugala, Kinga, Tuptuk, Nilufer, Hailes, Stephen, 2016. Evolving attackers against wireless sensor networks. In: GECCO 2016 Companion Volume, Denver, USA, 20-24 July 2016. ACM, p. 306.

Mrugala, K., Tuptuk, N., Hailes, S., 2017. Evolving attackers against wireless sensor networks using genetic programming. IET Wirel. Sens. Syst. 7 (4), 113–122.

Ojugo, A.A., Eboka, A.O., Okonta, O., Yoro, R.E., Aghware, F.O., 2012. Genetic algorithm rule-based intrusion detection system (gaids). J. Emerg. Trends Comput. Inf. Sci.

Pastrana, S., Orfila, A., Ribagorda, A., 2011. A functional framework to evade network ids. In: 2011 44th Hawaii International Conference on System Sciences, pp. 1–10.

Patton, R.J., Kambhampati, C., Casavola, A., Zhang, P., Ding, S., Sauter, D., 2007. A generic strategy for fault-tolerance in control systems distributed over a network. Eur. J. Control 13 (2), 280–296.

Rangaiah, Gade Pandu, Feng, Zemin, Hoadley, Andrew F., 2020. Multi-objective optimization applications in chemical process engineering: tutorial and review. Processes 8 (5).

Ricker, N.L., 1995. Optimal steady-state operation of the Tennessee Eastman challenge process. Comput. Chem. Eng. 19 (9), 949–959.

Ricker, N. Lawrence, 1998. Tennessee Eastman Challenge Archive. Available: http://depts.washington.edu/control/LARRY/TE/download.html#Multiloop.

Riquelme, N., Von Lücken, C., Baran, B., 2015. Performance metrics in multi-objective optimization. In: 2015 Latin American Computing Conference (CLEI), pp. 1–11.

Rush, George, Tauritz, Daniel R., Kent, Alexander D., 2015. Coevolutionary agent-based network defense lightweight event system (candles). In: Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation, GECCO Companion '15. ACM, New York, NY, USA, pp. 859–866.

Service, Travis, Tauritz, Daniel, 2009. Increasing infrastructure resilience through competitive coevolution. New Math. Nat. Comput. 05 (02), 441–457.

Service, T., Tauritz, D., Siever, W., 2007. Infrastructure hardening: a competitive coevolutionary methodology inspired by neo-darwinian arms races. In: Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International, vol. 1, pp. 101–104.

Symantec, 2011a. W32.duqu: The precursor to the next Stuxnet (version 1.4). Technical report, Symantec Security Response. Available: https://docs.broadcom.com/doc/w32-duqu-11-en. (Accessed 16 November 2023).

Symantec, 2011b. W32.stuxnet dossier (version 1.4). Technical report, Symantec Security Response. Available: https://www.wired.com/images_blogs/threatlevel/2011/02/Symantec-Stuxnet-Update-Feb-2011.pdf. (Accessed 16 November 2023).

Van Veldhuizen, D.A., Lamont, G.B., 2000. On measuring multiobjective evolutionary algorithm performance. In: Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512), vol. 1, pp. 204–211.

Vollmer, T., Alves-Foss, J., Manic, M., 2011. Autonomous rule creation for intrusion detection. In: 2011 IEEE Symposium on Computational Intelligence in Cyber Security (CICS).

Wang, Wei, Cammi, Antonio, Di Maio, Francesco, Lorenzi, Stefano, Zio, Enrico, 2018. A Monte Carlo-based exploration framework for identifying components vulnerable to cyber threats in nuclear power plants. Reliab. Eng. Syst. Saf. 175, 24–37.

Wei, Lu, Traore, Issa, 2004. Detecting new forms of network intrusion using genetic programming. Comput. Intell. 20 (3), 475–494.

Xia, T., Qu, G., Hariri, S., Yousif, M., 2005. An efficient network intrusion detection method based on information theory and genetic algorithm. In: PCCC 2005. 24th IEEE International Performance, Computing, and Communications Conference, 2005.

Zitzler, Eckart, Thiele, Lothar, 1998. Multiobjective optimization using evolutionary algorithms - a comparative case study. In: Eiben, Agoston E., Bäck, Thomas, Schoenauer, Marc, Schwefel, Hans-Paul (Eds.), Parallel Problem Solving from Nature - PPSN V. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 292–301.

Zitzler, Eckart, Deb, Kalyanmoy, Thiele, Lothar, 2000. Comparison of multiobjective evolutionary algorithms: empirical results. Evol. Comput. 8 (2), 173–195.

Zitzler, Eckart, Laumanns, Marco, Thiele, Lothar, 2001. Spea2: Improving the strength Pareto evolutionary algorithm. Technical report. Swiss Federal Institute of Technology (ETH).

Nilufer Tuptuk received her PhD degree in Computer Science from University College London in 2019. She currently works as an Assistant Professor in the Department of Security and Crime Science, University College London. Her research interests include cyber-physical systems security and the application of Artificial Intelligence in cybersecurity and cybercrime.

Stephen Hailes is a Professor of Wireless Systems and the Head of the Department of Computer Science at University College London. Stephen's research interests have three main directions: trust and security, in which he was one of the founders of the field of computational trust; networking, in which he has contributed to the development of the next-generation Internet and to the security of networked industrial control systems; and mobile, robotic and sensor systems. He has published extensively in journals and conferences in areas including AI and machine learning, robotics and automation, information security, behavioural ecology and sociology, gas sensor design and environmental monitoring.