Performance evaluation of cyber-physical intrusion detection on a robotic vehicle

Tuan Phan Vuong, George Loukas, Diane Gan
Department of Computing and Information Systems
University of Greenwich
London, UK
Email: (t.p.vuong, g.loukas, d.gan)@gre.ac.uk

***Abstract*—Intrusion detection systems designed for conventional computer systems and networks are not necessarily suitable for mobile cyber-physical systems, such as robots, drones and automobiles. They tend to be geared towards attacks of different nature and do not take into account mobility, energy consumption and other physical aspects that are vital to a mobile cyber-physical system. We have developed a decision tree-based method for detecting cyber attacks on a small-scale robotic vehicle using both cyber and physical features that can be measured by its on-board systems and processes. We evaluate it experimentally against a variety of scenarios involving denial of service, command injection and two types of malware attacks. We observe that the addition of physical features noticeably improves the detection accuracy for two of the four attack types and reduces the detection latency for all four.**

***Index Terms*—Cyber-physical systems; Mobile robots; Intrusion detection; Decision tree; Cyber-physical attack; Network security; Denial of service (DoS); Command injection; Malware; Detection Latency**

I. Introduction

Robotic vehicles, unmanned aerial vehicles (UAVs) and automobiles feature a tight coupling between their cyber and physical properties. We use the term cyber to refer to their computation and communication processes, and physical to their mobility, power consumption and any other physical manifestation of their operation. Our focus here is on examples of cyber-physical attacks, where a security breach in cyberspace has an adverse effect in physical space [1] (in the vehicle's physical function). Examples may include causing a vehicle to stop, a UAV to land prematurely [2], or its controls to be hijacked, as in the case of Maldrone [3], an experimental backdoor server for quad-copter drones, which kills the autopilot, takes control and can spread to other drones. Earlier examples include the work of Checkoway et al. [4] on a malware-infected audio file that was able to allow the researchers remote control of an automobile's engine control and braking system, and the work of Kerns et al. [5] in altering a vehicle's trajectory via GPS spoofing. Beyond these organised research efforts, there are also hacking competitions with time-restricted challenges for hijacking vehicles, such as the Tesla S sports car hijacking contest reported in [6].

We have previously identified that, depending on attack type and implementation approach (including fail-safe system and other automated behaviours), a robotic vehicle under attack on its communication channel can misbehave. It may head towards the wrong direction, be delayed, be forced to shut down, continue blindly, physically jitter [7] etc. Here, we take a more detailed look on the physical impact of different cyber attacks, including denial of service, command injection, and two types of malware infection. We also explore the feasibility of an intrusion detection system that takes into account physical input features, in addition to cyber input features, to detect these attacks reliably and rapidly. Our focus is on input features that can be measured by the robotic vehicle's resource-limited on-board components and monitoring processes. The aim is to provide the vehicle with the capability to detect attacks without relying on the knowledge or processing of any external system.

II. Related Work

Intrusion detection in cyber-physical systems is a relatively new area of research [8], which tends to focus on industrial control systems, such as programmable logic controllers (PLCs) and supervisory control and data acquisition (SCADA) systems. The range of related research that is applicable to robotic vehicles is still rather limited. One approach that has been proposed is to base the detection on whether sensor values collected and utilised by a robot are realistic. For instance, the intrusion detection system of the iRobot Create robotic platform needs to first determine what sensor values are "normal" and from these try to determine whether network traffic is suspicious [9]. In most cases, some form of network monitoring for detecting attempts for control hijack or modification is necessary, whether host-based as in the robotic surgery arm discussed in [10], or a

This work was supported by the UoG VC's PhD grant "Cyber-physical security of robotic vehicles"

hybrid host-based and network-based, as in [11]. In [12], detection is conducted both locally and collaboratively within a group of autonomous robots, by taking into account reputation, behaviour scores and distance. For distributed systems, which include multiple mobile elements, as in emergency response or battlefield situations, one can use a voting-based system, such as the one proposed in [13].

In general, detection can be categorised as behaviour-based, typically independent of type of attack, e.g. based on Petri Nets [14], or based on the knowledge of what impact a particular attack has on a particular robot, as in [7]. Our focus here is on a standalone robot, which does not have the opportunity to share information or collaborate with other robots to detect attacks. Examples could include remote-controlled standalone robots for reconnaissance, surveillance, rescue or bomb disposal missions. As a first attempt, we opt for a knowledge-based approach with supervised training where the robot learns a variety of attack types and is then challenged to detect attacks of similar nature. That is because in the first instance our aim is to determine whether an attack can indeed be detected reliably by the robot itself and whether the combined use of both cyber and physical input features is beneficial. In the near future, we will also develop behaviour-based approaches for comparison.

III. Testbed and Experimental Setup

The testbed for our development and experimentation is a four-wheel-drive robotic vehicle (Figure 1) controlled via an on-board Intel Atom computer running the Linux operating system and an Arduino micro-controller for driving the motors. The robot also carries a camera with pan and title functionality for situational awareness and remote navigation. The robot and its camera can be controlled remotely via an Ethernet cable or Wi-Fi, by relaying commands received over a TCP socket on the robots control board. Standard magnetic encoders fitted to the two rear wheel motors provide information on the angular position of each wheel. The difference between two consecutive measurements is representative of its speed. Figure 2 presents the components of the robot in more detail. It also shows the input features collected for the purpose of intrusion detection, identifying also the components from where they are collected.

Different attacks have different impacts on the computation, communication and physical operation of the robot. In particular the physical impact is not only an adverse effect of a cyber-physical attack, but also an opportunity. We argue that by monitoring these physical manifestations it is possible to improve the performance of a system designed to detect cyber attacks against a robot. To test the feasibility of this, we need to simplify the experimental conditions. First of all, the robot does not move in a real environment, but is placed on a stand

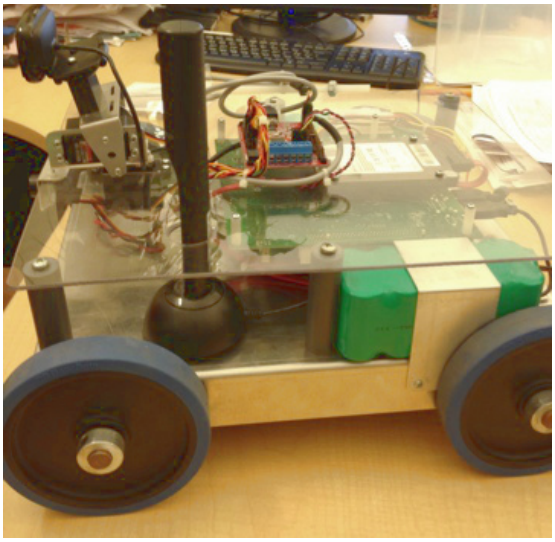

Fig. 1. Photo of the robotic vehicle

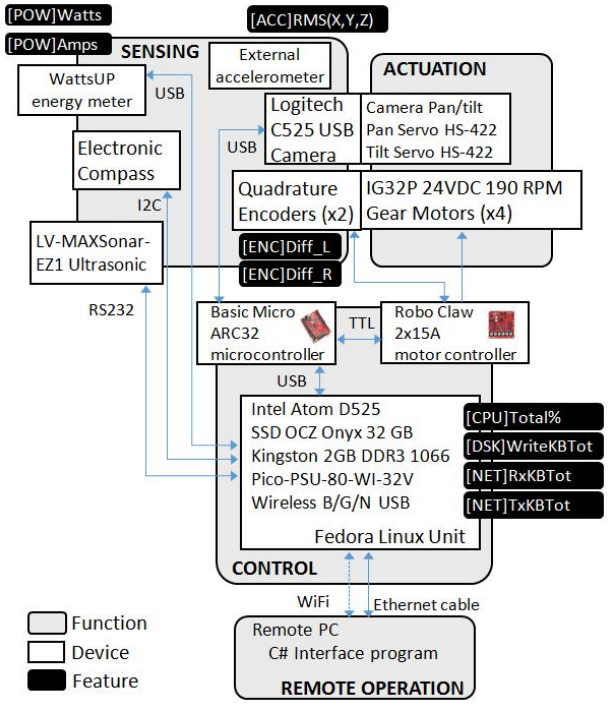

Fig. 2. Detailed diagram of the testbed. The input features used for the detection are shown in black background

during the experiments. This is to reduce the impact of any unrelated environmental effect, such as friction, and also helps with repeatability of scenarios in a consistent manner. Also for consistency across different scenarios, the robot is powered via DC power supply rather than its on-board battery pack. This ensures that the variable quality of battery packs does not affect the measurements, especially with regards to the performance of the magnetic encoders.

A. Attack vectors

As representative of a wide-range of possible attacks against a robotic vehicle, we have conducted experiments where the robot is under denial of service (DoS) attack, command injection attack, and two kinds of malware attack, one targeting the CPU and the other targeting the network (NET), as shown in Table I, which also summarises the primary physical impact observed during each one.

TABLE I
EXPERIMENTAL SCENARIOS

S#	Type	Impact observed
S1	DoS	Inconsistent stops
S2	Command Injection	Frequent consistent jittering
S3	Malware (NET)	Frequent consistent stops
S4	Malware (CPU)	No clear physical effect
S5	Normal operation	No adverse effect

1) Denial of Service attack (DoS) (S1): Here, the aim of the attack is to flood the robot's network interface with TCP traffic. The attack is a simple denial of service attack at a rate of approximately 8.7 MBit/s. The primary aim is to disrupt the communication between legitimate operator and robot. Intermittent connectivity causes the robot to trigger temporarily its fail-safe mechanism, which is simply to stop, and then resume its movement.

2) Command injection attack (S2): The robot receives commands from its legitimate operator to move forward, and at the same time receives rogue "stop" or "turn left" commands from an attacker.

3) Malware attack against network (Malware (NET))(S3): A piece of malware disrupts the network communication by causing a delay in the network. It utilises the Linux kernel's network scheduler to modify the network traffic control setting. As a result, the robot's movement becomes erratic with frequent stops during the attacks.

4) Malware attack against CPU (Malware (CPU)) (S4): A piece of malware consumes processing through a resource-demanding calculation. Unlike S1, S2 and S3, in this case no obvious change in the robot's physical behaviour is observed.

5) Normal operation (S5): Here, there is no attack. All network traffic and applications running are legitimate. They correspond to the camera feed transmitted to the operator, as well as the operator's legitimate commands to the robot.

B. Features

We focus on types of data that can be extracted by a mobile cyber-physical system without considerable overhead. We have identified eight input features, four related to communication and processing, which we refer to as the cyber input features, and four related to the physical properties of the robot, which we refer to as the physical input features. The attack label is the ground truth for the scenario, which corresponds to whether an attack really is present or not at a specific point in time.

- **Network Incoming:** Received network traffic rates.
- **Network Outgoing:** Transmitted network traffic rates.
- **CPU:** The total CPU utilisation.
- **Disk Data:** The rate of data being written to the disk.
- **Encoder:** Represents the wheel speed;
- **Accelerometer:** Represents the vibration of the chassis (using accelerometer measurements).
- **Power:** Corresponds to power consumption.
- **Current:** Corresponds to current.
- **Attack label:** This is the ground truth label, which states whether there is an attack or not at a particular point in time. This is used to train the intrusion detection system and also to evaluate its performance.

IV. CYBER-PHYSICAL INTRUSION DETECTION

Our goal here is (i) to provide a light-weight intrusion detection mechanism that can detect a cyber attack against a robotic vehicle using both cyber and physical input features and (ii) to compare the effectiveness of the intrusion detection against four different cyber attack types: DoS, Command injection, Malware against Network and Malware against CPU based on a performance metric including detection latency. As a lightweight approach, we used a decision tree learning algorithm for automatically producing detection rules that will be used by the robotic vehicle.

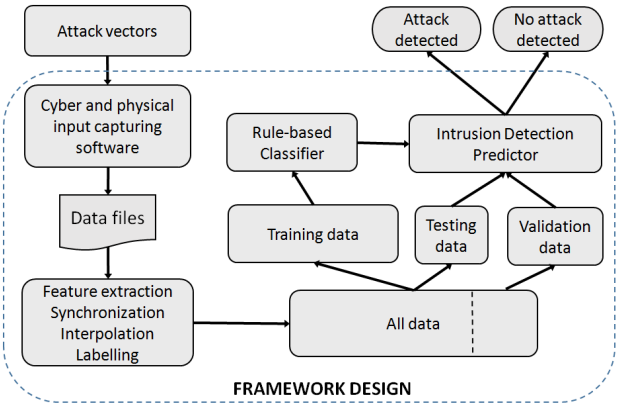

Fig. 3. Intrusion Detection Framework

A high level overview of the system is illustrated in Figure 3. Our framework design of the intrusion detection mechanism aims to run on the actual robot based on its own monitoring processes and components without relying on any external system. We have opted for a rule-based approach, because it is light-weight at run-time. For the generation of the rules, we use a decision tree machine learning algorithm for its speed

and simple construction. Machine learning is common in intrusion detection systems for conventional computer systems [15] (but still not for vehicles or other mobile cyber-physical systems). Especially the C4.5 decision tree algorithm has been used extensively to detect denial of service and other attacks [16]. Here, we use the improved C5.0 algorithm [17], [18], [19] with live data collected from the robotic vehicle. Before applying the C5.0 detection mechanism, we start with a data collection and preparation phase (Section IV-A).

A. Data collection and preparation

For each cyber attack scenario shown in Table I, the data for the cyber and physical input features are captured from different locations within the architecture (Figure 2) and at different points in time due to lack of perfect synchronisation and different sample collection periods (T) (see Table II). For example, the encoder value is collected by monitoring scripts embedded within the robot control unit, while Watts and Amps are measured with the WattsUp device [20]. As a result, the data needs to be processed to address the synchronisation difference between the clocks of these different data collection devices. Also, linear interpolation is used to address the fact that different devices would collect data at different time intervals. Figures 4, 5, 6 and 7 show representative runs for each of the scenarios S1, S2, S3 and S4 using the data after clock synchronisation and interpolation in R. We set the "ground truth" label to true when there is an attack and false when there is no attack. In total, the five scenarios present 52,215 data points for each feature.

TABLE II
CYBER (C) AND PHYSICAL (P) FEATURES AND THEIR COLLECTION PERIOD

Feature name	Description and Type (C/P)		Period (T)
RxKBTot	Network receive (KB)	C	1.0 s
TxKBTot	Network transmit (KB)	C	1.0 s
CPU	Total CPU usage (%)	C	1.0 s
WriteKBTot	Disk Write Data (KB)	C	1.0 s
DiffEncoderL	Change in Left Encoder	P	30 ms
RMS	Vibration of chassis	P	20 ms
Watts	Power consumption (W)	P	1.0 s
Amps	Electric Current (A)	P	1.0 s
Label	Attack Flag (1,0)		1.0 s

B. Training, testing and validation data for each attack

As mentioned already, S5 corresponds to the normal operation of the robot. Each attack (S1-S4) includes both the legitimate activities in the normal operation of S5 and the illegitimate activity introduced by the particular attack. For the purpose of training, testing and validation, we split the data (Figure 3) into two sets with sample size of:

- 80% for training and testing, of which the training data is chosen randomly from the 70% of this division, and testing is chosen from the remaining 30%.
- 20% for validation. This is holdout data that is not seen by the training models.

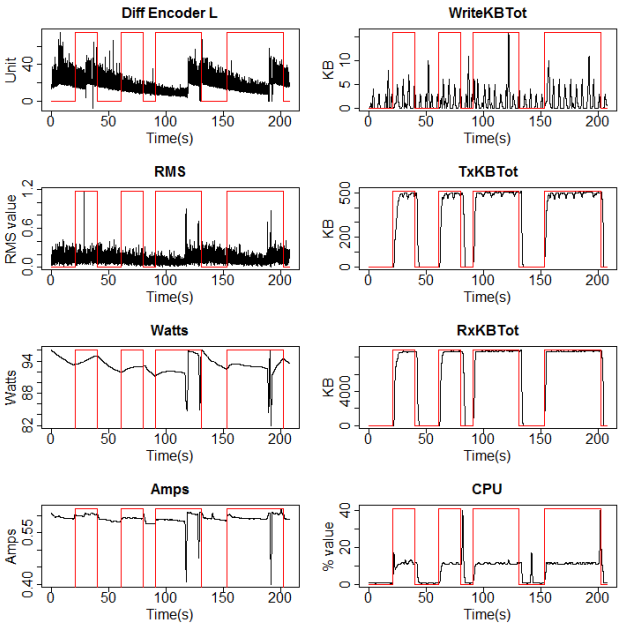

Fig. 4. The data for cyber and physical features collected during the denial of service attack (S1). The overlaid frames denote the periods of time that the denial of service attack is on.

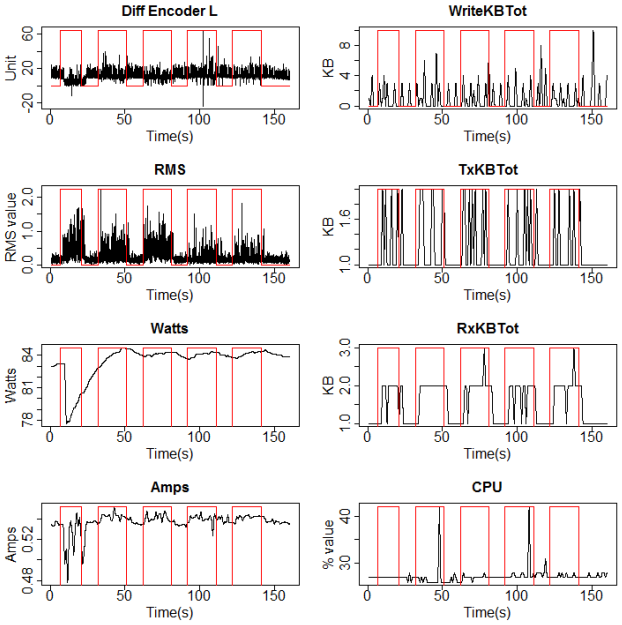

Fig. 5. The data for cyber and physical features collected during the command injection attack (S2). The overlaid frames denote the periods of time that the command injection attack is on.

C. Detection method

We have used the decision tree C5.0 package [21] in R to generate the rule-based classifier. Each Decision Tree model is fit by Quinlan's C5.0 algorithm for each

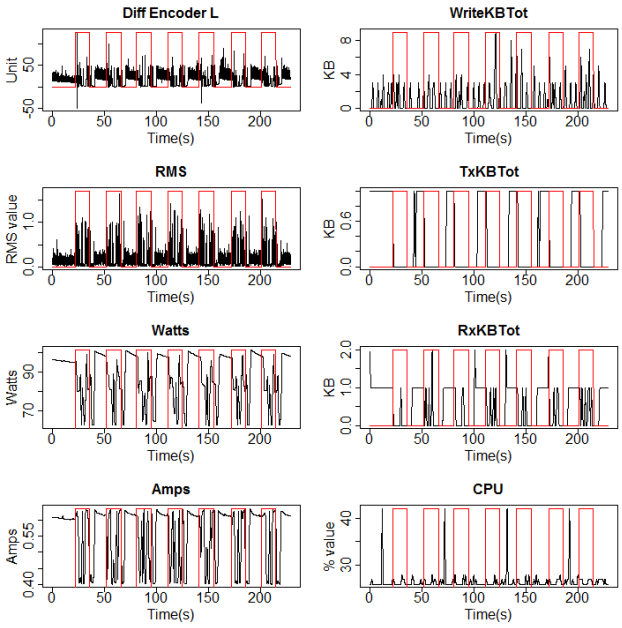

Fig. 6. The data for cyber and physical features collected during the malware attack against Network scenario (S3). The overlaid frames denote the periods of time that the network malware is active.

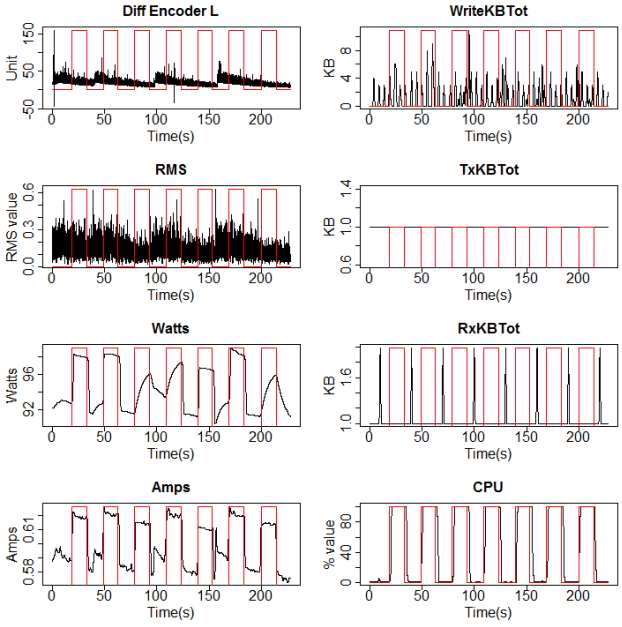

Fig. 7. The data for cyber and physical features collected during the malware attack against CPU scenario (S4). The overlaid frames denote the periods of time that the CPU malware is active.

different training data as mentioned in Section IV-B. So, it creates one model (one ruleset) for each attack type.

Then the intrusion detection classifier of each attack is evaluated against the testing data and the validation data separately. This evaluation is in terms of its ability to correctly recognise the existence or absence an attack at each point in time. A sample section of the decision tree rules generated is shown in Figure 8.

```
Decision tree:

Amps <= 0.6098701:
:...Amps <= 0.5962737: 0 (9802/3)
:   Amps > 0.5962737:
:   :...Watts <= 92.19859: 1 (18)
:       Watts > 92.19859:
:       :...WriteKBTot <= 3.892: 0 (172)
:           WriteKBTot > 3.892:
:           :...CPU <= 2.032: 0 (4)
:               CPU > 2.032: 1 (8)
Amps > 0.6098701:
:...Amps <= 0.613997:
    :...Watts > 96.03431: 0 (35)
    :   Watts <= 96.03431:
    :   :...CPU <= 3.376004: 0 (9/2)
    :       CPU > 3.376004: 1 (155)
    Amps > 0.613997:
    :...Watts <= 97.85741: 1 (555)
        Watts > 97.85741:
        :...Watts > 98.1: 1 (545)
            Watts <= 98.1:
            :...Watts <= 97.9:
                :...WriteKBTot <= 0.01599979: 1 (42)
                :   WriteKBTot > 0.01599979: 0 (23)
```

Fig. 8. An example of the Decision Tree rules generated

V. Evaluation

Choosing performance metrics for intrusion detection in cyber-physical systems is not trivial, because their priorities are different to those of conventional computer systems. Here, we use the confusion matrix, receiver operating characteristic (ROC) curves with its area under the curve (AUC) and detection latency.

A. Confusion matrix

The confusion matrix relates to the number of errors in the outcome of the intrusion detection. The rate of false positives ($FPR = FP/(FP+TN)$) and false negatives ($FNR = FN/(FN+TP)$) with regards to the "ground truth" and the overall accuracy rate ($ACC = (TP+TN)/(TP+FP+TN+FN)$) are appropriate for the evaluation of the performance of our system. As cyber-physical systems are still not common targets of attacks, they are likely to be the target of non-standard and possibly zero-day attacks. This makes FNR important. In fact, FNR may also affect the detection latency.

As mentioned in Section IV-B, the experiment consisted of building four different detection models for the different attack types.

TABLE III
Detection results using only cyber input features

	Test	Validation		
Attack	**ACC%**	**FPR%**	**FNR%**	**ACC%**
DoS	99.45	15.77	7.26	90.47
Command inj.	97.58	31.79	22.34	72.80
Malware (NET)	94.99	21.42	18.99	79.70
Malware (CPU)	97.03	21.16	6.76	85.31

Tables III and IV show high accuracy rates above 94% for testing data, which reflects the "local-optimal"

TABLE IV
DETECTION RESULTS USING BOTH CYBER AND PHYSICAL INPUT FEATURES

	Test	Validation		
Attack	**ACC%**	**FPR%**	**FNR%**	**ACC%**
DoS	99.84	10.76	41.44	66.70
Command inj.	99.53	29.60	5.74	81.99
Malware (NET)	99.20	25.70	11.31	80.92
Malware (CPU)	99.72	5.43	26.18	85.24

feature of the decision tree C5.0 algorithm. This is because the training and testing data are sharing very similar characteristics as they are from the same set (but different samples). So, the testing data is the ideal condition where the attack observed is very similar to the attack the system has been trained on. The results using validation data correspond to the more realistic case, where the attack is of the same type but not identical. Detection results for the DoS attack are relatively poor (albeit better than the random guess). The other three attack detection models provide considerably better results with regards to ACC.

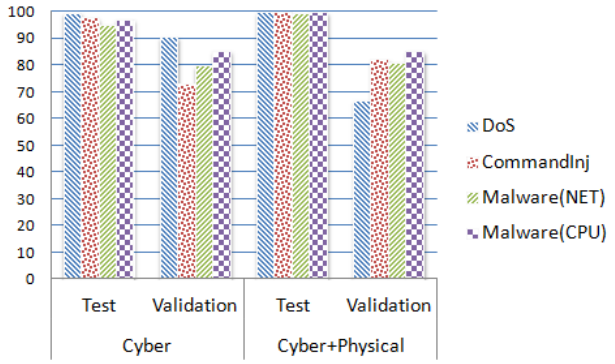

Fig. 9. Accuracy chart for the four models on Test and Validation data using Cyber only and Cyber+Physical features.

B. Receiver Operating Characteristic Curves

TABLE V
AUC COMPARISON USING CYBER ONLY AND BOTH CYBER AND PHYSICAL INPUT FEATURES

	AUC	
Attack	**Cyber only**	**Cyber + Physical**
DoS	0.89	0.73
Command inj.	0.75	0.87
Malware (NET)	0.82	0.86
Malware (CPU)	0.91	0.97

As our system is effectively a binary classifier ("Yes, there is an attack" vs. "No, there is no attack"), we also use ROC curves to measure its performance. The ROC curve plots the true positive rate (TPR) against the FPR for different thresholds as shown in Figure 10. In an ideal detection result, the curve should go through the point (0,1) where the FPR is 0% and TPR is 100%. Area under the curve (AUC) of the ROC curves is a metric to compare the quality of different binary classifier models.

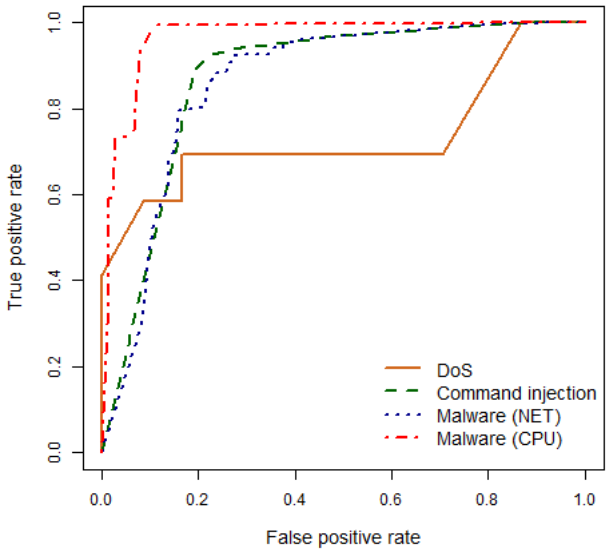

Fig. 10. The ROC curves of the detection rules for the four attacks: DoS, Command Injection, Malware (NET), Malware (CPU) and all, in the case where all eight cyber and physical input features are utilised.

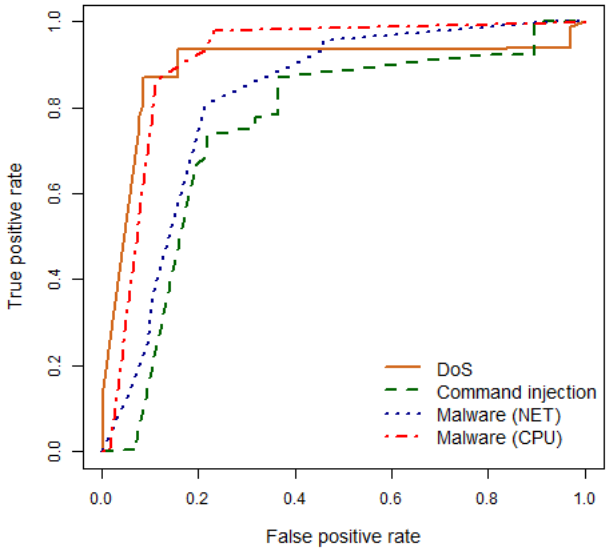

Fig. 11. The ROC curves of the detection rules for the four attacks: DoS, Command Injection, Malware (NET), Malware (CPU) and all, in the case where only the four cyber input features are utilised.

The performance of all four detection models is demonstrated in Figure 10. Using the ROCR package in R [22], we can visualise how TPR and FNR change for

each detection model with different probability thresholds. As can be observed, the AUC for the malware attack against the CPU (0.97) is considerably higher than other attack scenarios. The models for command injection and malware (NET) attack have similar AUC at 0.87 and 0.86 respectively. The DoS scenario has the lowest AUC model at 0.73.

Repeating the same experiments without taking into account the physical input features, the performance of the intrusion detection system drops noticeably for command injection and the malware attacks, but not for DoS (see table V for the detailed comparison).

C. Detection latency (DL)

A primary concern for cyber-physical systems is their real-time nature, which means that timeliness, and hence detection latency is particularly important. In fact, detection latency may be potentially more important than the accuracy of the detection, as there is no point in detecting an event after it has caused permanent physical damage to a vehicle (e.g., by causing it to veer off the road and crash). A few researchers have included detection latency as a metric, but mostly in relation to mobile ad hoc networks and wireless sensor networks [23], [24]. We propose that this is a significant metric for assessing the performance of our system. This has also been recognised in Mitchell and Chen's excellent survey in [8].

Based on the design of the intrusion detection framework (Figure 3), there are various factors that affect DL including the data collection time, the processing time and the actual detection accuracy of the mechanism.

TABLE VI
DETECTION LATENCY (MS) FOR DIFFERENT ATTACK TYPES (CYBER ONLY VS. CYBER + PHYSICAL)

	Attack block (s)			Detection latency	
Attack	Block	Start	End	C (ms)	C+P (ms)
DoS	B1	374.04	423.04	1020	1000
Command inj.	B2	312.32	331.32	2020	1460
	B3	342.32	361.32	2340	1040
Malware (NET)	B4	362.02	376.02	2020	1940
	B5	393.02	407.02	1520	1000
	B6	422.02	436.02	2020	2020
Malware (CPU)	B7	360.06	374.04	2020	1200
	B8	390.06	404.04	1000	1000
	B9	420.7	435.04	1000	1020

The data collection time is the time it takes for all data to reach the detection system. It depends on the collection period (T) of each feature, as in Table II, the time taken for data preparation, including interpolation period, and the time it takes the data to be sent for processing. Then, there is the processing time, which is the time it takes the algorithm to process the data and reach a detection decision.

The data collection and processing times remain largely the same across different detection scenarios

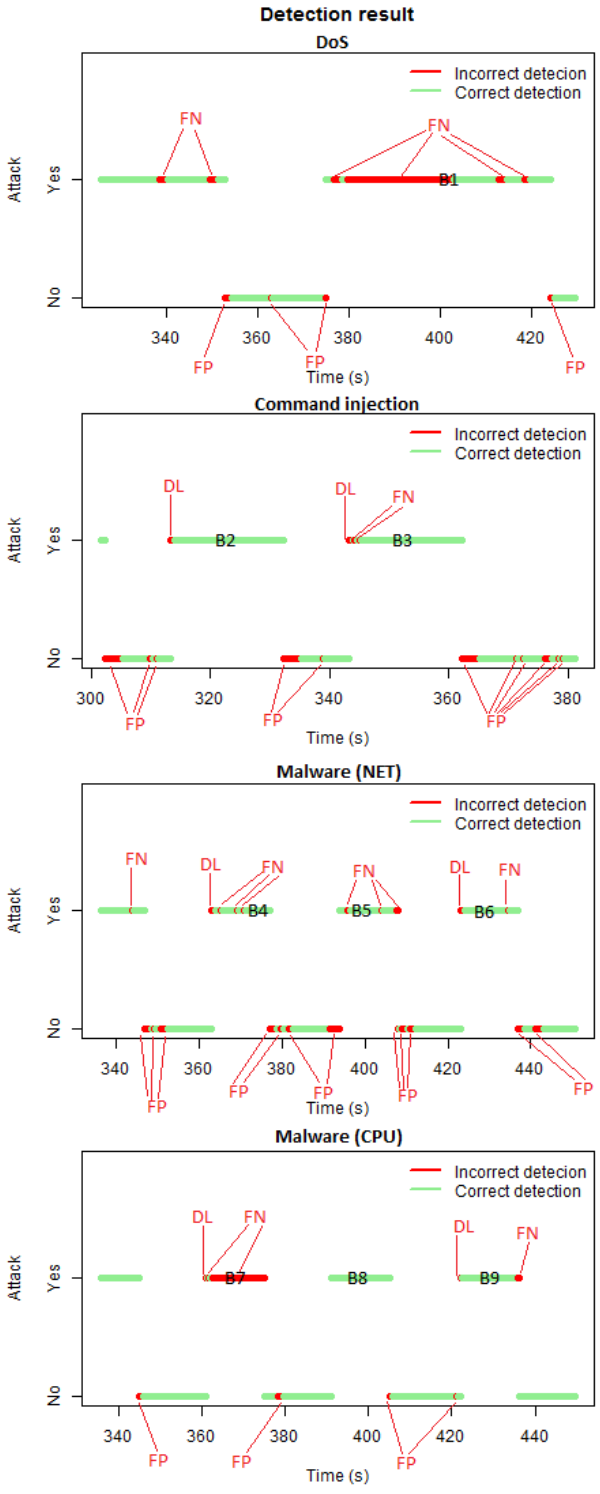

Fig. 12. Detection result for representative attack scenarios

(around 1 s). What does differ significantly is the delay in relation to the accuracy of the algorithm. A false negative outcome would delay the detection of an attack until the first true positive result is achieved for a given attack. So,

a visual presentation of detection latency is depicted in Figure 12. Note that the YES Vs. NO points correspond to the ground truth. So, an incorrect detection on a YES line is a false negative (FN), and on a NO line is a false positive. Detection latency is added by the time of the first FN block (DL block) occurring at the beginning of each YES (attack time) block (see Table VI).

VI. Conclusion and Future Work

As real-world and experimental cyber-physical attacks are becoming more prevalent, onboard intrusion detection systems are expected to become particularly important, especially for standalone robotic vehicles, whether remote-controlled or autonomous. We have hypothesised that the existence of physical manifestations of cyber attacks on vehicles and other cyber-physical systems constitutes an opportunity for intrusion detection purposes. We have experimented with four different types of attack and have observed different performance for each one. For example, we see that utilising physical features in addition to cyber features can increase the false negatives dramatically, but this is not the case for command injection attacks. For these, the addition of physical features improves all detection performance metrics. Similarly improved performance is observed for the two types of malware attacks.

Looking beyond traditional performance metrics, what does appear to be a consistent benefit is the noticeably lower detection latency. We argue that for robotic vehicles this is important. Poor accuracy, such as high false positives, is expected in some cases, especially for highly dynamic systems that are influenced by their physical environment. Here, the detection latency can be more important, for example, if an unacceptably high delay means that a cyber-physical attack can cause the vehicle to crash and injure human beings before it is detected.

We now intend to extend the scope of our work by experimenting with more attack types, such as communication jamming and replay attacks, as well as with different detection mechanisms. A decision tree-based approach can be very light-weight, but is not flexible enough to address attacks that the system has not been trained on. For this purpose, in addition to this knowledge-based approach, we are also working towards testing the hypothesis that the addition of physical input features can improve not only the detection latency but also the accuracy of a behaviour-based detection approach.

References

[1] G. Loukas, *Cyber-Physical Attacks: A Growing Invisible Threat*. Butterworth-Heinemann (Elsevier), 2015.

[2] G. Jennings, "Iran claims to have flown reverse-engineered us stealth uav," November 09, 2014. London - IHS Jane's Defence Weekly.

[3] R. Sasi, "Maldrone the first backdoor for drones," January 26, 2015. Fb1h2s aka Rahul Sasi's Blog.

[4] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *Usenix Security Symposium*, 2011.

[5] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys, "Unmanned aircraft capture and control via gps spoofing," *Journal of Field Robotics*, vol. 31, pp. 617–636, 2014.

[6] J. Griffiths, "Zhejiang University team scoops 10,600 Yuan for hacking into Tesla Model S," July 17, 2014. South China Morning Post.

[7] T. Vuong, A. Filippoupolitis, G. Loukas, and D. Gan, "Physical indicators of cyber attacks against a rescue robot," in *IEEE International Conference on Pervasive Computing and Communications*, pp. 338–343, IEEE, 2014.

[8] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 55, 2014.

[9] A. S. Uluagac, V. Subramanian, and R. Beyah, "Sensory channel threats to cyber physical systems: A wake-up call," in *2014 IEEE Conference on Communications and Network Security (CNS)*, pp. 301–309, IEEE, 2014.

[10] T. Bonaci, J. Herron, T. Yusuf, J. Yan, T. Kohno, and H. J. Chizeck, "To Make a Robot Secure: An Experimental Analysis of Cyber Security Threats Against Teleoperated Surgical Robotics," pp. 1–11, 2015.

[11] S. Shetty, T. Adedokun, and L.-H. Keel, "Cyberphyseclab: A testbed for modeling, detecting and responding to security attacks on cyber physical systems," 2014.

[12] A. Fagiolini, G. Dini, and A. Bicchi, "Distributed intrusion detection for the security of industrial cooperative robotic systems," in *World Congress*, vol. 19, pp. 7610–7615, 2014.

[13] R. Mitchell and I.-R. Chen, "Adaptive Intrusion Detection of Malicious Unmanned Air Vehicles Using Behavior Rule Specifications," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, p. 1, 2013.

[14] V. Skormin, A. Dolgikh, and Z. Birnbaum, "The behavioral approach to diagnostics of cyber-physical systems," in *AUTOTESTCON, 2014 IEEE*, pp. 26–30, IEEE, 2014.

[15] K. Sravani and P. Srinivasu, "Comparative study of machine learning algorithm for intrusion detection system," in *Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2013*, pp. 189–196, Springer, 2014.

[16] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1690–1700, 2014.

[17] B. R. Patel and K. K. Rana, "A survey on decision tree algorithm for classification," *International Journal of Engineering Development and Research*, vol. 2, 2014.

[18] S. Noh, C. Lee, K. Choi, and G. Jung, "Detecting distributed denial of service (ddos) attacks through inductive learning," *Intelligent Data Engineering and Automated Learning*, pp. 286–295, 2003.

[19] A. Filippoupolitis, G. Loukas, and S. Kapetanakis, "Towards real-time profiling of human attackers and bot detection," in *Proceedings of 7th International Conference on Cybercrime Forensics Education and Training (CFET)*, 2014.

[20] Watts up? meters, www.wattsupmeters.com.

[21] M. Kuhn, W. Steve, and N. Coulter, "Package C50," 2014.

[22] T. Sing, O. Sander, N. Beerenwinkel, and T. Lengauer, "Rocr: visualizing classifier performance in r," *Bioinformatics*, vol. 21, no. 20, pp. 3940–3941, 2005.

[23] M. Striki, K. Manousakis, D. Kindred, D. Sterne, G. Lawler, N. Ivanic, and G. Tran, "Quantifying resiliency and detection latency of intrusion detection structures," in *Military Communications Conference, 2009. MILCOM 2009. IEEE*, pp. 1–8, IEEE, 2009.

[24] T.-L. Chin and W.-C. Chuang, "Latency of collaborative target detection for surveillance sensor networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 26, no. 2, pp. 467–477, 2015.