2015

Information fusion architectures for security and resource management in cyber physical systems

Brijesh Kashyap Chejerla

Department: Computer Science

INFORMATION FUSION ARCHITECTURES FOR SECURITY AND RESOURCE MANAGEMENT IN CYBER PHYSICAL SYSTEMS

by

BRIJESH KASHYAP CHEJERLA

A DISSERTATION

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

2015

Dr. Sanjay Madria, Advisor
Dr. Ali Hurson
Dr. Sriram Chellappan
Dr. Wei Jiang
Dr. Maciej Zawodniok

PUBLICATION DISSERTATION OPTION

This dissertation consists of five articles prepared in the style required by the journals or conference proceedings in which they were published:

Pages 11 to 42, “Efficient Spatio-Temporal Information Fusion in Sensor Networks”, was published in 14th IEEE International Conference on Mobile Data Management (MDM) 2013 Milan, Italy.

Pages 43 to 88, “An Information Fusion Architecture for Secure and Robust Wireless Networked Control System”, submitted to IEEE Transactions on Cybernetics.

Pages 89 to 124, “An Information Fusion Architecture for Scheduling under Variable-load in a Cloud-Integrated Cyber Physical System”, was submitted to 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID 2015), Shenzhen, Guangdong, China.

Pages 125 to 163, “Robust Task Scheduling and QoS in Attack Resilient Cloud-assisted Cyber Physical Systems ", was submitted to 2015 35th IEEE International Conference on Distributed Computing Systems (ICDCS 2015), Columbus, Ohio, USA.

A journal version of the same paper - Pages 125 to 163“Attack Resilient Scheduling in Dynamic Cloud-Assisted Cyber Physical Systems” will be submitted to Journal of Grid Computing, Springer.

ABSTRACT

Data acquisition through sensors is very crucial in determining the operability of the observed physical entity. Cyber Physical Systems (CPSs) are an example of distributed systems where sensors embedded into the physical system are used in sensing and data acquisition. CPSs are a collaboration between the physical and the computational cyber components. The control decisions sent back to the actuators on the physical components from the computational cyber components closes the feedback loop of the CPS. Since, this feedback is solely based on the data collected through the embedded sensors, information acquisition from the data plays an extremely vital role in determining the operational stability of the CPS. Data collection process may be hindered by disturbances such as system faults, noise and security attacks. Hence, simple data acquisition techniques will not suffice as accurate system representation cannot be obtained. Therefore, more powerful methods of inferring information from collected data such as Information Fusion have to be used.

Information fusion is analogous to the cognitive process used by humans to integrate data continuously from their senses to make inferences about their environment. Data from the sensors is combined using techniques drawn from several disciplines such as Adaptive Filtering, Machine Learning and Pattern Recognition. Decisions made from such combination of data form the crux of information fusion and differentiates it from a flat structured data aggregation. In this dissertation, multi-layered information fusion models are used to develop automated decision making architectures to service security and resource management requirements in Cyber Physical Systems.

ACKNOWLEDGEMENTS

The journey of a Ph.D. student is a long one where impressionable young minds mature into rational, cognitive, analytical and influential minds. In my journey, I have met many people who acted as a catalyst of varying degree in my maturity.

First and foremost, I would like to express my deepest gratitude towards my advisor Dr. Sanjay Madria for his continuous support and patience during my Ph.D. study and research. I have greatly benefited from his knowledge, advice and guidance, which was instrumental in the completion of this dissertation.

I would like to specially thank Dr. Sriram Chellappan for his invaluable motivation and advice at the start of my Ph.D. which immensely helped me stay focused on my long term goals. I would also like to thank the members of my dissertation committee, Dr. Ali Hurson, Dr. Wei Jiang and Dr. Maciej Zawodniok, for their constructive comments and feedback.

I would like to acknowledge the contributions of the current and former members of my research group. I would like to especially single out Dr. Vimal Kumar whose support and friendship have been invaluable. I would like to give a special mention to Late Sri Harsha Chitturi, with whom I would have had a life long friendship, if it were not for his untimely demise. I am profoundly thankful to my friends Dr. Deepak Balaji, Dr. Ravi Akella, Srikanth Beerla, Dr. Prathyush Reddy, Ramesh Chembeti and Dr. Phalgun Lolur, who, during the course of my Ph.D. made my life memorable.

I owe everything I have achieved to my father Mr. Vijaya Kumar, my mother Mrs. Bala Tripura Sundari and my sister Bindu Madhavi. I thank the universe for them. Many people have had an influence on my life, but none more profound than Dr. Neelanjana Dutta who encouraged, challenged, motivated and inspired me at every stage of my Ph.D. I dedicate my dissertation to these four people.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Page

PAPER I

PAPER II

LIST OF TABLES

LIST OF ALGORITHMS

1. INTRODUCTION

In the domain of distributed systems, information acquisition and processing has vital importance in enabling accurate system representation to maintain operability via sustained reliability and robustness. Cyber Physical Systems (CPS) are such a breed of distributed systems with a tight coupling between collaborating computational elements controlling physical entities. In a CPS, embedded computers and networks monitor and control the physical processes, with feedback loops where physical processes affect computations and vice versa. Some applications of CPS include but are not limited to Networked Control Systems, Transportation Systems, Power Plants etc. Each of these CPSs have either hard or soft real-time Quality of Service (QoS) constraints imposed by the criticality of the information collected through the sensing components embedded in the system.

CPSs comprise of a network of sensors embedded to the control plant which perform sensing and data acquisition. Inferring information from the collected data is performed on local or or remote computing resources. However, there may be inaccuracies in the data collected from the sensors owing to the presence of system faults or security attacks. This will lead to an inaccurate representation of the system information further leading to inefficient usage of the CPS resources. Since vulnerabilities exist in both physical system and computing resources, multi-sensor information fusion to tackle security and resource management is vital in maintaining the system operability.

The discipline of multi-sensor information fusion, distributed sensing and data acquisition has rapidly evolved over the years to solve a set of diverse problems arising from the increasing need of accurate system state representation in CPSs. Information fusion combines data/ information from different sensors to give a global view of the system that is being observed. This is in contrast to data acquisition from a set of individual sensor which gives limited information, thereby, restricting inference about the features of data. Data

from different sources is processed and combined using methods from several disciplines such as adaptive filtering, statistical estimation, pattern recognition, machine learning, regression analysis, network analysis and queuing theory.

As opposed to the general view where data aggregation and information fusion are considered one and the same, information fusion extends to decision making in addition to combining raw data obtained from the sensors. There are various levels of information fusion which ultimately aid in automated decision making capabilities. They are:

- Level 1 processing - *object assessment* - Fusion of multi-sensor data to determine attributes, characteristics and identity of the observed entity.
- Level 2 processing - *situation assessment* - Automated reasoning to determine the relationships among the observed entities and the environment and the meaning of the information obtained about the observed entities and generate hypotheses.
- Level 3 processing - *impact assessment* - Predicting and assessing the impact of the information at hand in a future time step or window to generate an alternative hypothesis other than the existing one to determine possible threats or vulnerabilities.
- Level 4 processing - *process refinement* - Regressive and recursive information fusion in order to improve the accuracy of the information already inferred.
- Level 5 processing - *cognitive refinement* - Interaction between the data fusion system and human decision maker to improve the interpretation of results.

A detailed description of the manner of usage of the multiple levels of information fusion on data collected from the sensing components deployed in various applications that have been researched in this dissertation is described.

The first chapter of this dissertation focuses on Wireless Sensor Networks which are comprised of a network of sensors deployed in various topologies for sensing and data reporting. Typically, sensor participation is limited to passive sensing and reporting of the

collected data to the base station, where, information obtained from the data is processed offline. There are various methods of data forwarding schemes such as flooding, hierarchical routing, leader election etc. Of these, hierarchical routing methods are most commonly used in WSNs. Due to the topology of sensor deployment in the different applications of WSNs, energy conservation of the power sources of sensors becomes extremely crucial. In order to tackle the challenges of energy consumption, in-network data aggregation schemes have been extensively proposed. However, in spite of the various improvements, data aggregation is still bottlenecked by the frequency of messages communicated and the need for tight sensor synchronization. In order to alleviate this problem, a spatio-temporal information fusion scheme which uses Level 1 of the information fusion was developed in paper I. The scheme takes into consideration the fact that energy consumption during communication is higher than that under computation by several orders of magnitude. An adaptive filtering technique is used to reduce computation each time new data is sensed. Spatio-temporal information is used to equip each sensor with the global knowledge of the system state. This is advantageous in contrast to passive sensing and reporting where, data corresponding to events and sensor faults cannot be quickly and accurately classified. Through the information fusion scheme, data fusion is still possible even under weak sensor synchronization owing to the global or local system knowledge of each sensor in the topology. These advantages are highlighted by the overall energy reduction reflecting in effective resource management of the WSNs.

The applications of WSNs extend to sensing and data acquisition in CPSs. However, in CPS applications, the sensors cannot be abstracted from the physical system. They are embedded and integrated in the functioning of the physical system of the CPS in real-time. Hence, they cannot simply be treated as stand alone sensing devices, but as sensing devices that have to cater to the changing Quality of Service (QoS) needs of the physical system. Information about the sensing requirements is fed to the sensors through the control decisions made by the cyber system in the CPS application. Along with the sensing information, the

cyber system also sends information about the control decision to close the control feedback loop to the physical system. This entire process makes the CPS a much more dynamic and complex domain. Increasing number of subsystems in the CPS increase the operational complexity. Therefore, the servicing cyber system must be equipped with an efficient architecture that can handle the changing QoS requirements of the physical system and the embedded sensor network. The biggest challenge in maintaining the QoS requirements will be under the influence of security attacks. Traditional CPSs comprise of wired sensing devices that are connected to the remote processing units via internet. In such scenarios, providing a strong firewall against attacks is sufficient. Recently, many organizations and companies are considering the implementation of wireless devices connected to the base station which is then connected to the remote processing units to reduce implementation and operational costs. In such implementations, the attack space of the attacker intending to launch security attacks on the system will be high.

Most common application of a CPS can be found in Networked Control Systems (NCSs). NCSs using a wireless communication are called in Wireless Networked Control Systems (WNCSs). Paper II of this dissertation takes into consideration the security aspects that govern the NCS stability and proposes an information fusion architecture which uses a novel Time-Varying Dynamic Bayesian Network (TVDBN) as the learning mechanism. A feature extraction and selection algorithm is proposed along with the Gaussian Mixture Models to classify the various features of the NCS. Paper II uses levels 1, 2 and 3 of the information fusion to improve the decision making capabilities. The major contribution of this work is that of proposing an multi-level inference architecture which probabilistically determines the overall effect of stand-alone and collaborative attacks through effective feature selection. This eliminates the handicap existing security mechanisms in NCS and WNCS that heavily rely on cryptographic techniques.

As the scope of CPS is increasing everyday there is an exponential growth in complexity owing to an increase in the number of subsystems of the CPS. The research chal-

lenge with such dynamic and complex systems is to tightly couple the physical and cyber components through available physical attributes and acquired information. The solution to this challenge is a decision support system through information fusion which tracks, predicts and addresses the needs of the physical system by outputting control decisions which closes the feedback loop of a CPS. Such a decision support system cannot be made available through the on-board processors on the physical components or the existing computing infrastructure which is static. Therefore there is a need for a more scalable infrastructure such as provisioned by Cloud Computing. This new outlook of the CPS is called cloud-assisted CPS. Paper III of this dissertation focuses on this research problem by proposing an information fusion architecture that fuses information obtained from a feature selection algorithm to build a causal graphical model. This graphical model is then used to allocate resources appropriately through a dynamic scheduling algorithm. The proposed scheduling algorithm ensures that even under variable and bursty incoming traffic, the resources of the cyber system are not over utilized while constantly maintaining a bounded latency which is one of the QoS requirements of the CPS. Paper III uses Levels 1, 2, 3 and 4 of information fusion.

Next in paper IV of this dissertation, the research extends to the aspect of robust scheduling of cyber resources under security attacks, where the cyber system is cloud. Human intervention and interaction during security attacks have not been adequately addressed in the domain of a CPS. Hence, inputs regarding human behavior, decisions and their scope and extent must be taken into consideration while developing scheduling algorithms that satisfy the QoS requirements of CPS. To address this problem, a new approach to semi-network form game theory to obtain a robust and attack resilient scheduling mechanism for a cloud assisted CPS was proposed. The research progresses into studying the importance of serving mixed-criticality of the scheduled tasks on the cyber resources. Mixed-criticality CPS are systems which comprise of both dependent and independent tasks whose priority is determined by their overall impact on stability and reliability. Paper IV uses Lev-

els 1, 2, 3 and 4 and 5 of information fusion.

Throughout the research conducted in this dissertation, information fusion is used across multiple levels to provide automated decision making capabilities to the applications considered. The dissertation is organized in the order of papers produced while reflecting the logical flow of the increased complexity and scope of research.

2. LITERATURE SURVEY

Real world applications of Cyber Physical Systems extend to various domains such as but not limiting to: *Supervisory Control and Data Acquisition systems (SCADA), Smart Grids and Transportation systems*. However, the requirements and Quality of Service (QoS) parameters for each of the applications varies in nature of the topological setting and the communication environment. Hence, the primary aim of the cyber component aiding such dynamic systems is to maintain system operational stability under changing environments such as security attacks, and QoS requirements such as latency, availability and robustness. In order to satisfy these QoS requirements, resource allocation plays a major role. Allocating both serviced and serving components to the cyber physical domain in a real time environment is a challenging task which at most times has no optimal solution. Owing to the ever increasing heterogeneity of the CPS applications, large volumes of data are being generated. Serving such large data can no longer be done on the traditional computing resources that exist. This brings in the need for a scalable computing platform such as the cloud.

In this dissertation, the main topics that have been tackled are (i) Security of CPS (ii) Scheduling of mixed criticality CPS tasks under both normal and variable loads. Therefore, in order to familiarize the reader with the existing work in the field, we present the literature survey in two domain aspects of CPS that have been mentioned.

2.1. SECURITY IN CPS

There are primarily two types of security attacks in CPS: passive and active. Furthermore, passive attacks have two types of behavior - releasing the message contents and traffic analysis (using methods such as packet sniffing). Passive attackers view the contents of the messages that they are not authorized for, or, in the case of traffic analysis, they

observe the traffic patterns and use inferring techniques to determine the network behavior. On the other hand, active attacks comprise of attack styles that include - masquerade attacks where a signal is sent to the network from a seemingly valid user, who is in actuality an attacker; Replay or deception attacks e.g. false data injection attack, where the attacker captures the signal or data value at the previous time instant and repeatedly sends the same signal or data value to destabilize the CPS or steal resources from the CPS; and Denial of Service (DoS) attacks which are the most common types of attacks. In a DoS attack, service to the users can be denied either on the physical component or on the cyber component depending on the attackers incentive needs.

In [1] deception and DoS attacks against an NCS are introduced, and, for the latter attack, a countermeasure according to semi-definite programming is proposed. In [2, 3], false data injection attacks against static state estimators are introduced and studied. It is shown that undetectable false data injection attacks can be designed even when the attacker has limited resources. In a similar fashion, stealthy deception attacks against the SCADA systems are studied, among others, in [4, 5]. In [6], the effect of covert attacks against networked control systems is investigated. In [7], a resilient control problem is studied, in which control packets transmitted over a network are corrupted by a human adversary. This kind of attack is an insider attack. A receding-horizon Stackelberg control law is proposed to stabilize the control system despite the attack. Recently, the problem of estimating the state of a linear system with corrupted measurements has been studied [9]. More precisely, the maximum number of faulty sensors that can be tolerated is characterized, and a decoding algorithm is proposed to detect corrupted measurements. Finally, security issues of some specific cyber-physical systems have received considerable attention, such as power networks, linear networks with misbehaving components and water networks [3, 7].

Distributed DoS (DDoS) is another variant of the DoS attack performed on distributed components of the CPS. It has a sufficiently long recorded history of attacks and warrants a mention of all the existing work in relation to the work in this dissertation. Mirkovic and

Reiher proposed a taxonomy of DDoS attack and defense mechanisms [10], and Peng et al. presented a survey of network-based DoS attacks and defense techniques [11]. Game theory has been previously applied to gain insights into cyber security issues. Roy et al. surveyed game-theoretic solutions to network security applications, largely along the line of the types of games used (i.e., static and dynamic games) and whether the information available to the players is perfect or imperfect, and complete or incomplete [12]. Manshaei et al. surveyed previous works on applying game-theoretic techniques to address security and privacy problems [30]. Previously, there were a few efforts on conducting game-theoretic analysis of DDoS attacks and defense. Zang et al. applied a Bayesian game model to analyze the defense against DDoS attack traffic with unclear signatures [14]. In their model, the defender is uncertain about the type of the traffic origin, which can be either a legitimate user or an attacker, and thus infers it using Bayesian rules. In [13], Snyder et al. introduced a DDoS traffic injection game, which is a two-person zero-sum game with imperfect knowledge. In the model developed by Wu et al. [15], the attacker attempts to optimize the attack effect by choosing the most effective attack traffic sending rate or number of zombie machines to send out attack traffic, while the defender optimizes the effectiveness of filtering attack traffic at the firewall. The entire game works in a continuous setting and the Nash equilibrium strategy can be computed analytically. In [16], Xu and Lee proposed a defense system against DDoS attacks and analyzed its performance in a game-theoretic context. Khirwadkar et al. developed a repeated game model based on the fictitious play process for pushback-based DDoS defense [18]. The key idea of their approach is that each player estimates the mixed strategy of the opponent's actions based on her previous observations, and then plays the pure strategy that is the best response accordingly. In [17], Yan and Eidenbenz proposed a truthful mechanism to provide economic incentives to ISPs to defend against DDoS attacks in a non cooperative environment. Besides pushback and firewall filtering, client puzzle is another DoS/DDoS defense mechanism, in which a client has to solve a computational puzzle before the server commits

resources to deal with his or her request. In [19, 20], Game theory has also been applied to study this type of protection against DDoS attacks.

2.2. SCHEDULING OF MIXED CRITICALITY CPS TASKS

Mixed-criticality systems are the subject of much recent research due to the emergence of cyber-physical systems. The US Air Force Research Laboratory has been leading a Mixed Criticality Architecture Requirements (MCAR) [26] initiative to investigate building blocks to safely construct such mixed criticality systems. An extensive literature on real-time systems exists regarding the study of offline overload scheduling problem [23, 28]. These studies are focused on uni-processor systems, and try to maximize the accrued value from task completions. Researchers have also looked at online scheduling of overloads [21, 25, 27]. These schemes were not designed for mixed criticality systems and do not include an explicit notion of criticality, and hence cannot take advantage of this notion. Approaches such as the elastic scheduling model [22] deal with overloads in a criticality-aware fashion by allowing tasks with higher elasticity to run at higher rates when required, whereas tasks with lesser elasticity are restricted to more steady rates. Authors in [24] have developed the Zero Slack Rate-Monotonic Scheduling (ZSRM) algorithm for mixed-criticality scheduling on a single processing setting. ZSRM provides asymmetric temporal protection guarantees under which low-criticality tasks cannot interfere with high criticality tasks but high-criticality tasks can steal cycles from low-criticality tasks under overload scenarios to meet their deadlines. Scheduling tasks on a set of processors is a well-studied problem in the context of real-time systems. Traditional solutions are classified into (i) global scheduling, (ii) partitioned scheduling, and (iii) semi-partitioned scheduling [29]. Among these solutions, partitioned scheduling algorithms incur the least run time cost. Such cost includes task state migration, number of rescheduling instants, and a larger execution time of such rescheduling.

PAPER

I. EFFICIENT SPATIO-TEMPORAL INFORMATION FUSION IN SENSOR NETWORKS

Brijesh Kashyap Chejerla, Sanjay K Madria

Department of Computer Science,

Missouri University of Science and Technology, Rolla, Missouri 65401

Making the sensor data look more meaningful in its representation of an observed entity is the primary goal of sensor data fusion. Due to the energy constraint on sensors, there exists a need for algorithms that minimize the fusion cost while maintaining the validity of the data sent to the base station. Maintaining validity is even more difficult when we have a limited knowledge of the factors that govern an observed sensor entity. To achieve this goal, we modeled the uncertainties in sensor data and fed them into the system, employing recursive data estimation. By doing so, we considered the dynamically changing environmental parameters affecting the network to produce the most accurate representation of the observed system state. We propose here a spatio-temporal, correlation-based estimation procedure to corroborate the detection of an event in a sensor field. The number of in-network communications plays a great role from the networking perspective. This is because the power consumption during communication is several times greater than the power consumption during computation. To achieve this, our algorithm ensures that communication is done only during the time of an event. At all other times, the sensor motes maintain an updated global estimate, without communicating, by using a prediction algorithm. This reduces the need for frequent sensor synchronization. We conducted experiments using our distributed fusion architecture to show our algorithm's effectiveness; by a reduction in the power consumption, in terms of both the computation and the communication.

I. INTRODUCTION

Wireless Sensor Networks collect data from sensor-monitored applications. Combining this data into a meaningful representation of the feature being monitored is referred as data/information fusion. Many methods employ either a centralized or a distributed architecture to fuse data [3, 5, 8, 10]. Of these, the distributed scheme is becoming increasingly relevant to systems that are ad-hoc by nature as the parallel nature of this architecture makes it more robust and fault-tolerant. In sensor networks, the distributed architecture allows for the calculation of globally fused estimates from the local estimates by using an appropriate fusion criterion. The nodes have a good understanding of how the neighbors report data. This understanding can also be useful in detecting not only faulty nodes but also outliers.

Many statistical techniques and signal processing methods have been proposed for the purpose of data fusion [1, 2, 3, 4, 5]. These estimation schemes are generally proposed for implementation on a centralized architecture. Their flexibility then becomes an issue in many practical scenarios. C. J. Ran and Z. L. Deng in [3] realized the correlation of estimation errors in a two-sensor structure. Kalman filtering [1] is one common approach used in a distributed architecture that uses both estimation and prediction to obtain fused estimates based on a suitable fusion algorithm. As sensors operate in environments surrounded by uncertainties, they are capable of producing missing/corrupted data. Therefore, the fusion algorithm must be both fault-tolerant and accurate enough to work with restrained resources. An important issue that governs the accuracy of a system under observation is the noise input to the state estimation. Most Kalman filter applications assume the noise is a known a-priori and uncorrelated. This assumption is the major drawback in works that use Kalman filter for fusion. In practice, the process noise behavior is unknown; it might be correlated with the measurement noise. In general, errors induced in the estimation are accounted for the noise in the system under observation. Regression allows us to obtain

a good estimate from one filter. The use of multiple such filters to obtain accurate global estimates in uncertain environments poses new problems in this direction. Fusing information and not just data is, therefore, a challenge in a multi-sensor environment, where each sensor could be affected differently by noise. Our scheme differs from other fusion schemes in this aspect. In other works [13], in-network data aggregation is done to reduce the communication cost. Various levels of information fusion, such as feature extraction and decision-making, were not included. They were considered to be a separate part in the fusion algorithm (i.e., most pre-processing and post-processing work is done offline). Logically, this was done to reduce the computational overhead on the sensors. However, our algorithm allows this to be done on the sensors so that they can take part in the decision-making process without increasing the computational overhead.

Estimation is perhaps the primary purpose for using the Kalman filter. State prediction is done based on the one-step prediction process. This process is generally done in a static environment and thus does not deal with the effects of a varying noise. This prediction can be extended and the filter can be made to predict several steps ahead to understand the behavior of the system under study. Such a use of the Kalman filter in data fusion techniques, in which, the decision process is performed on the motes, has not, to our knowledge been applied previously.

Large networks require large matrices. The computation of a matrix inversion within these large networks becomes a burden on the aggregating node. On the contrary, our proposed method uses only the estimates that are bound to have a high variance with both the previously observed value (temporal) and the reference global estimate (spatial). If the previous estimate is accurate (i.e. the error is within the bound), the observed value is considered as is. By doing so, we can reduce the number of times state estimation is done. This process can be predetermined by the predicted values that tell us how a filter is going to behave.

In our algorithm, we made use of both the temporal and the spatial correlations in

determining which sensor data to use in the fusion process. The fusion rule we employed uses the correlations among various sensors that report different parameters. Dynamically updating the filter characteristics, has not, to our knowledge been explored previously for sensor data fusion. Estimation of the noise also plays an important role in obtaining accurate state estimates. Changing noise patterns are calculated periodically. Both the process noise and the measurement noise covariance are updated in the filter implementation.

Our approach also reduces computational complexity by avoiding the use of complex matrices. Only the values that are correlated to a sensor are taken into consideration, thus keeping both the covariance and transition matrices small. Our aim was to reduce not only the error and process covariance as much as possible but also reduce the convergence time. This reduction will give us accurate estimates quickly in order to detect an event.

To prove the effectiveness of our algorithm, we ran simulations of an underground mine in which environmental factors changed the observed value. This simulation was done using FLUENT, based on an experiment conducted at NIOSH [14]. The simulation took into consideration how the inflow of gases in a mine gob is affected by environmental parameters, such as atmospheric pressure. A correlated noise file was generated offline. The values were input to the filter to simulate the dynamic changing behavior of a real-time application.

Our results indicate that our algorithm allows the system to dynamically adapt to the changes in noise by updating the filter with new parameters. The filter predicts the values at the pre-determined n^{th} time step in advance, sending the value deemed the reference value for a time slot. As a result the number of communications is reduced greatly. Time slots were calculated based on previous observations. In critical event detection situations such as detecting mine fires in an underground mine, predictions are important to allow prior knowledge at a later stage for reference and to take appropriate action. Our scheme shows good results as the fire is predicted based on the increase in the concentrations of combustible gases. By this, mine fires are proactively prevented rather than reactively

detected. This predicted value is sent to the base station. The worst-case scenario, in which the prediction must be sent very frequently is naturally avoided, as such events do not occur very often.

Our approach is also fault tolerant as each sensor in the network has the global estimate. This estimate is updated regularly through both spatial and temporal information. The time the next set of transmissions for new global estimate calculation must be sent is computed based on this information. This reduces the need for continuous synchronization among the sensors, thereby reducing the number of transmissions required. Hence, this ensures that, if a good throughput is obtained, that throughput will reduce the communication cost, as no retransmission of the same information occurs.

The rest of the paper is organized as follows. Section II talks about the existing works in this field and gives a comparison to our paper. Section III gives a brief introduction to Kalman filter. In section IV we give a formal description of the problem solved. Section V describes our approach in detail. In section VI we detail our fusion algorithm. Section VII shows the experimental setup and we analyze our results and finally, in Section VIII we conclude the paper.

II. RELATED WORK

A number of data fusion schemes have been proposed on wireless sensor networks. Fusion and aggregation are the same with two exceptions: information fusion and decision fusion. Data fusion, for applications like target tracking, use state estimation schemes that provide good estimates about the system state at a particular time. These estimations are then fused based on mathematical techniques, such as min max, averaging, average weighted estimates and regression [3]. These averaging techniques are used in applications in which a certain degree of leniency is not only handled but is also acceptable. These fusion/aggregation schemes, when applied to wireless sensor networks, need a great deal of both modification and accuracy. This is because of the time critical and mission critical applications in which the sensors are deployed. In many real world applications, we do not have prior knowledge. In such cases, both estimation and prediction techniques such as Kalman filtering [1] have been used extensively. The basic Kalman filtering technique is linear. It is the only Kalman filtering technique that is computationally cheap (as compared to other techniques of its kind). Thus, a number of fusion techniques have been built upon its capability [2, 3, 4, 5, 8, 9, 10].

Olfati-Saber [8] proposed a distributed Kalman filtering algorithm. This algorithm uses the consensus filters that allow for the calculation of average consensus in time-varying signals. We used the Kalman filtering scheme to obtain estimates from several sensors by giving weights to each observation. Each sensor acted as a micro-Kalman filter whose estimates are the local states. These are then fed into the global estimator. Li et.al and Ran et.al [2, 3] propose methods, where, each method calculates the optimal state estimates by using the prediction capabilities of the Kalman filter. They considered one step state predictions only. The computations are very heavy as the filters use complex matrices. For situations in which the inverse of a matrix is to be computed, the computational cost on the

sensor is very high, thereby draining the energy available.

Abdelgawad et.al [10] proposed a method in which the sensor readings are quantized to one bit so as to reduce the communication costs. This method can work for sensor deployments in which quantized values are not a cause for concern. Again, although it has a very effective way of communicating among nodes, the algorithm is very flat. It does not include decision-making abilities embedded into the algorithm. Khan et.al [5] discussed reducing the number of sensors' readings included in the matrices by presenting a graph-based solution. This solution works effectively for sensors with less correlation. The computational matrix, however, may remain large for correlated sensor values in a very large sensor network deployment. Various researchers [7, 13] have discussed the fusion of sensor data with regard to communication constraints in the channel, thereby reducing the total number of messages sent. This in turn reduces the communication costs and energy used. The fusion is done as per the effectiveness of performing a fusion scheme during the implementation of a routing procedure. Authors in [12] talk about data fusion from a controls perspective. This perspective is a layered approach that uses information from heterogeneous sensors. The higher-level sensors send feedback to the lower-level sensors that acts as a validation mechanism. All of these approaches fail to consider the changing parameters. Our aim was to develop an algorithm that makes the sensors decide whether or not to send a sensed value. This is shown in the subsequent sections.

III. A BRIEF INTRODUCTION TO KALMAN FILTER

A state is a mathematical representation of a physical system as a set of both input and output variables. Many scholars have worked towards improving the state space representation, developing many estimation techniques in the process. The Kalman filtering technique [1] is the most commonly used method on the merit of its accuracy and ease of use. It is a regression technique capable of producing quickly converging accurate states, even with minimal prior information.

A regression technique is one in which the typical value of the dependent variable changes when any one of the independent variables is varied, while, the other independent variables are held fixed. In figure 1, we show the depiction of how a Kalman filter works in its entirety. Here, $\hat{x}^{-}$ is the state of the system. $\hat{x}^{-}_{k-1}$ is the input to the system. P_k is the error covariance of the system. K_k is the Kalman gain of the system. Q and R are the process and measurement noise co-variances of the system [3], respectively, z is the measurement of the system, I is the unity matrix, A is an $n \times n$ transition matrix, and B is an $n \times n$ input matrix.

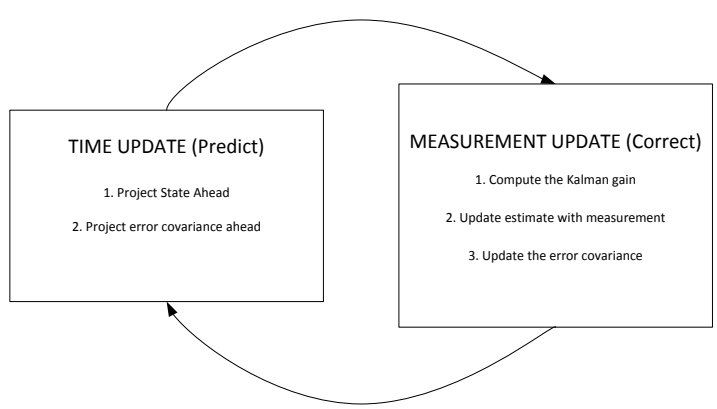

Figure 1. The Kalman filter update and correction steps

IV. PROBLEM FORMULATION

Sensors are deployed in areas where the noise is uncertain because of changing environmental variables. Any estimate of the noise is only an approximation of its possible behavior. We can determine the accurate readings from this noise model with little confidence. When these estimates are used in data fusion for feature extraction, these inaccuracies may only mislead us, thereby defeating the purpose of fusion.

Many false positives and false negatives will result in event detection applications. We focus here on one such event detection application, which is underground mine fire reporting. Reporting a mine fire when it occurs is perhaps most crucial in the mining industry. Several methods have been implemented for fire detection. Typically, however, it is reported after the fire has been detected and sometimes the detection is delayed. By using the prediction capability of the Kalman filter, we can estimate the time at which an event might take place, alerting the base station to prevent any hazards. We can then request help in advance by taking necessary measures. To this effect, we modeled the entire process as a linear time discrete system. This system is stochastic with unknown process noise.

$$x_k = Fx_{k-1} + Bu_{k-1} + w_{k-1} \qquad y_k = Hx_k + v_k$$

$$E(w_k w_j^T) = Q_k \delta_{k-j}$$

$$E(v_k v_j^T) = R_k \delta_{k-j}$$

$$E(w_k v_j^T) = S_k j \delta_{k-j}$$

Here, E is the expectation, or mean of the values. Q_k and R_k are the process noise covariance and measurement noise covariance respectively. S_k is the cross-covariance of the process and the measurement noise, w and v are the process and measurement noises

respectively, F is the transfer function, and x_{k-1} is the state of the system at the time instant k. δ_{k-j} is the Kronecker delta function. It is 1 if $k = j$, otherwise it is 0. B is a known matrix, u is the input to the system, and H is the matrix that is the orthogonal projection of y on x.

In order to solve the problem using Kalman filter, we have done the following:

- Found the predicted n^{th} step estimate of the system i.e. find $x_{k+n}||x_k$ from the state x_k.
- Built the unknown noise parameters S_k from the known parameters Q_k, R_k. Also, we built new estimates of the noise covariances Q_k and R_k and updated the system dynamically.
- Defined the fusion process that takes into account the correlations among the sensor parameters.

V. APPROACH OVERVIEW

We used the Kalman filter to estimate the noise that is input to the filter. In most cases, the measurement noise and the process noise are correlated to each other. In general although we do not have a strong understanding of how the process noise changes, we should be able to start off with a good idea about the measurement noise. We can determine the process noise based on the correlation between the two. Updating the Kalman filter with these noise parameters will keep the system state as close to the actual state as possible.

Estimated states of sensors are fused using the spatially correlated states multiplied by the individual cross-correlation coefficient and the inverse of the estimation error. This is done in a distributed environment (this is the 'fuse' mentioned in the algorithm). This value is then normalized to obtain the posteriori-fused estimate. Next, the nth step fused state estimate is computed. This estimate determines when the sensors will become active, come out of the sleep state, and send information. Doing so ensures that the sensors do not have to keep broadcasting their values at several time instants if the value of the observed state at that point of time is below a predetermined threshold value (which is calculated offline). This in turn ensures that a synchronization protocol is not a necessity. Event detection is done using both the time-correlated values of each sensor and the reference estimate. If the fused estimate value crosses the threshold, an alert message is sent to the base station for appropriate action. If the alert is a false positive, the Kalman filter is reset with new input parameters that more accurately reflect the changes in the environment.

A stepwise explanation of the proposed algorithm when the sensor is in an active mode is as follows:

1. Collect the data on each sensor mote in shorter time intervals.

2. Compute the state prediction up to the kth state using the n-step state prediction according to how dynamically the system varies.

3. To obtain accurate results, input the changing noise parameters into the filter as new variables that govern the system state.

4. Fuse estimates based on the weighting function, which is the product of the correlation (spatial and temporal) and cross covariance (spatial and temporal) of the sensors in communication and sensing range.

5. Use the fused estimate as the reference for event detection (based on a preset threshold). This is reported to the base station using relay messages.

Our algorithm can be extended to a dynamically changing system with mobile nodes. In our case, the implementation constraints boil down the algorithm to a static sensor deployment with a fixed group of sensors in a given area. This is consistent with the sensor network setup in a mine.

Our algorithm is written for the up time of the sensor mote, where the sensor is active. The sensor comes out of the SLEEP state based on a predefined manner in accordance with the synchronization mechanism explained above.

The sensor is put into the SLEEP state manually, to simulate the faulty behavior of a sensor. During this time, the sensor does not take any readings, and we can test for the fault tolerance of the sensor network setup. The following two subsections describe in detail how we compute the n-step state prediction, update the covariances Q and R, and build from the unknown noise parameters.

A. N^{TH} STEP STATE PREDICTION

Let us consider the case where the Kalman filter is used for one step prediction. The process and the measurement noises are one step correlated i.e. the measurement noise v_k

Algorithm 1: Dynamic Spatio-Temporal Information

```
Input: System state x̂_i^-, co-variances Q, R
Output: predicted state x̂_i^+
BOOLEAN: SLEEP
DEFINE: Preset Threshold
while TRUE do
    if !SLEEP then
        data ← x_i
        collect data and assign as new state x_i
        compute one step Kalman filter state prediction using Q and R
        Errorcovariance P̂_i^+ = min(P̂_i, P̂_i^-)
        P = P̂_i^+
        if x̂_i < preset threshold then
            compute n step state prediction
        else
            route x̂_n to the base station
            Update Q, R
            Fuse x̂_i ∈ Spatio − temporal correlated nodes
            /*fuse is done using the fusion process described later on*/
            Update x̂_i = x̂_{i_fused}
        end
        if x̂_{i_fused} > preset threshold then
            Send x̂_{i_fused}
            /*send to base station*/
        end
        SLEEP = TRUE
        /*send the motes to sleep state*/
    end
end
```

is correlated with w_{k-1}. The estimation errors are given as

$$\varepsilon_k^- = x_k - \hat{x_{k-1}}^- \tag{1}$$

$$\varepsilon_k^+ = x_k - \hat{x_{k-1}}^+$$

Where, $\hat{x_{k-1}}^-$ is the apriori one-step prediction at time k, and $\hat{x_{k-1}}^+$ is the posteriori. Now the classical Kalman filter state estimation is given as

$$\hat{x_k}^- = F_{k-1}\hat{x_{k-1}}^+ \tag{2}$$
$$\hat{x_{k-1}}^+ = \hat{x_k}^- + K_k(y_k + H\hat{x_k}^-)$$

The error can be computed, from the above equations as

$$\varepsilon_k^- = x_k - \hat{x_{k-1}}^- = F_{k-1}\varepsilon_k^+ + w_{k-1} \tag{3}$$
$$\varepsilon_k^+ = x_k - \hat{x_{k-1}}^+ = \varepsilon_k^- - K_k(H_k\varepsilon_k{}^- + v_k)$$

The priori and posteriori error covariance from [11] can be computed as

$$\begin{aligned} P_k^- &= E[\varepsilon_k^-(\varepsilon_k^-)^T] = F_{k-1}P_{k-1} + F_{k-1} + Q_{k-1} \\ P_k^+ &= E[\varepsilon_k^+(\varepsilon_k^+)^T] = P_k^- - K_kH_kP_k^- - K_kE[v_k(\varepsilon_k^+)^T] \\ &\quad - P_k^-H_k^TK_k^T + P_kH_k^TK_k^T - H_k^TK_k^T \\ &\quad + K_kE[v_k(\varepsilon_k^-)^T]H_k^TK_k^T - E[\varepsilon_k^-v_k^T]K_k^T \\ &\quad + K_kH_kE[v_k^T]K_k^T + K_kE[v_kv_k^T]K_k^T \end{aligned} \tag{4}$$

and, $E[\varepsilon_k^- v_k^T]$ can be computed as

$$E[\varepsilon_k^- v_k^T] = E[(x_k - x_k^-)v_k^T] = E[(F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1})v_k^T] - E[x_k^- v_k^T] = S_k \tag{5}$$

Because, the state x and the noise v are not correlated, the expectation is 0. Likewise, input u_{k-1} is also uncorrelated with v and the final term is also 0 as neither the priori nor

the next step measurement noises are correlated. Substituting $E[\varepsilon_k^- v_k^T]$ in the covariance (P_k^+) equation and re-arranging the terms by eliminating terms that become 0, gives us:

$$P_k^+ = (I - K_k H_k) P_k^- (I - K_k H_k)^T + K_k R_k K_k^T K_k (H_k S_k + H_k S_k^T) K_k^T - S_k K_k^T - K - K S_k^T \quad (6)$$

Based on the previously described technique, consider the case of n step correlation between the process and measurement noise. The expectation/mean can be stated as

$$E[w_k v_j^T] = S_{ij} \delta_{k-j+1} \forall i = 1, 2, ...n \quad (7)$$

Now, from the system state equations 1 we have

$$\begin{aligned} x_{k+1} &= F_k x_k + B u_k \\ x_{k+2} &= F_{k+1} F_k x_k + F_{k+1} B u_k \\ &\vdots \\ x_{k+n} &= F_{k+n-1} \cdots F_k x_k + F_{k+n-1} \cdots F_{k+1} B u_k \end{aligned} \quad (8)$$

From the process covariance, expanding the term $E[(F_{k-1} x_{k-1} + G_{k-1} u_{k-1} + w_{k-1}) v_k^T]$ similarly until the nth step the term, $(F_{k+n-1} \cdots F_k w_{k-1}) v_k^T$ and all other terms become 0; which gives us the covariance of the two correlated process. Substituting the above in the process noise covariance P_k^+ and expanding P_k^+ we can obtain the nth step covariance as

$$P_k^+ = P_k^- - K_k [H_k P_k^- + \sum_{j=1}^{n} \alpha_{jk} S_{jk} - \sum_{i=1}^{n-2} \beta_i * (\sum_{j=k+1}^{k+n-1} \alpha_{jk} S_{jk}) - \beta_{n-1} S_{nk}^{\ T}]$$

where, $\alpha_{jk} = F_{k+n-1} \cdots F_k$ and

$\beta = F_k(I - K_kH_k) * F_{k+1}(I - K_{k+1}H_{k+1}) * \cdots F_{k+n-1}(I - K_{k+n-1}H_{k+n-1}) * (F_{k+n}H_{k+n}K_{k+n})$

Hence, using the above equations we can find the nth step predicted state estimates. As the number of steps (after which predictions are required) increases, the error covariance adjusts accordingly. The Kalman filter gain, which plays a major role in the prediction process, is governed by both the measurement noise variance and the correlated noise covariance S_{kj}. By estimating the unknown process noise, the measurement noise, and the correlated noise, we can dynamically update the system state for better estimates.

B. BUILDING THE UNKNOWN NOISE PARAMETERS

Let us consider a system in which the two given noise parameters keep changing. In such a system, the general case of fixing noise as a constant will not suffice. The filter must be fed with the right noise parameters by using a method that dynamically estimates a change in noise, to obtain an accurate state estimation. Only then, will the filter behave accurately, remaining true to the system's observations with a reduced error.

In practice, before implementing the filter, noise parameters are fed into the state equations. These inputs might be close enough for the filter to begin producing expected results. We can input a high process error covariance value for cases in which the initial process noise is unknown, to start with. We can then run the filter until such time that the error converges to an accepted value. By that time, the measurement error values would also be optimal. We must also consider the correlation between the process noise and the measurement noise. For cases in which the measurement noise is unknown, the measurement error can pass as the measurement noise until we can calculate the measurement noise. From this correlation, we can better understand the noise parameters, how they behave, and how they affect the environment in which the sensors are deployed.

We can better understand how much the system will alter under the influence of a particular noise model, where the noise model represents the different input noise based on

the error covariance. We can then build a correlation function based on this understanding. This function defines the correlation between the process and the measurement noises.

For scenarios in which there is a sudden change in both the environment and the noise model, we propose computing the changed noise parameters by running the same process described above on spatially correlated sensors. Thus far we have calculated the temporally correlated noise of the system. A change alters the correlation coefficient, and this alteration cannot be estimated. Hence, we repeat the above process before taking the global noise covariance estimates. This is done by fusing the local noise covariance from each spatially correlated sensor. This ensures that the noise input to the filters is the one that is relevant to the area that a particular sensor is sensing.

Let us again consider the linear system 1. We can construct the unknown parameters as follows. Let us consider the inputs to the filter as the observations from other filters. Doing so allows us to build the correlated noise parameters. Using x for both the input values and the noise parameter produces

$$x_k - z^{-1}Fx_k = z^{-1}Bu_k + z^{-1}w_k x_k = (I - Fz^{-1})^{-1}Bz^{-1}u_k + (I - Fz^{-1})^{-1}z^{-1}w_k \quad (9)$$

These values correspond to the time k (time instant for which we are estimating the noise parameters). Substituting the value of x_k in $y_k = Hx_k + v_k$ produces,

$$y_k = H(I - Fz^{-1})^{-1}Bz^{-1}u_k + H(I - Fz^{-1})^{-1}z^{-1}w_k + v_k H(I - Fz^{-1}) = Bz^{-1}u_k + z^{-1}w_k + v_k H(I - Fz^{-1}) \quad (10)$$

Now, the error is,

$$(I - Fz^{-1})^{-1}Bz^{-1}u_k = z^{-1}w_k + v_k H(I - Fz^{-1}) \quad (11)$$

From this, we can find the covariance of this error as

$$E[((I-Fz^{-1})^{-1}Bz^{-1}u_k)(I-Fz^{-1})^{-1}(Bz^{-1}u_k)^T]$$
$$= E[(z^{-1}w_k + v_kH(I-Fz^{-1}))(z^{-1}w_k + v_kH(I-Fz^{-1}))^T]$$

This will help us find the correlation coefficient of the process and measurement noise. Thus, we can determine the unknown terms in the noise model using this scheme. Once we understand the unknowns, the filter must be updated with the new values for the covariance to obtain an accurate state estimation.

VI. SENSING, DATA ACQUISITION AND REPORTING

A. FUSION ALGORITHM

Previous studies that used the Kalman filter for fusion have used it only for estimation. They have then fused the estimated values by considering some weighted fusion approach. As most papers use the matrices in Kalman filters, using weighted matrices in the LMV (Linear Minimum Variance) sense is one of the most common approaches. Our approach tends to avoid the complex matrix calculations by considering only the values from the sensors that are both time and space correlated. Temporal autocorrelation is used to determine the relation several changing parameters have on the observed system state. This autocorrelation is also used to determine the threshold value. Spatial correlations are used in determining whether or not a particular sensor node participates in the fusion process. The advantage in using this approach is that we can avoid the complex matrix calculations while reducing the overall computational and communication complexity.

Our approach first fuses the data from several sensors. It then estimates the current state based on data from both multiple sensors as well as its own a-priori estimate. Naturally, the sensor gives utmost credit to its own reading. The other sensors' readings are fed as the inputs to this sensor (filter). The governing factor that determines the weight of the received value is a product of both the correlation and the inverse of the cross-covariance of the two filters. This cross-covariance is a result of taking into account the system's noise. We believe this approach gives the closest possible estimate of the observed state.

The spatial correlations in a static sensor network deployment will not change greatly unless there is a sensor that reports false data (as observed globally by a group of sensors). Under such scenarios, the correlation of that particular sensor ceases to exist with those of the group to which it belongs. This process can be done either before sensor network

deployment or during the information gathering stage.

Our algorithm also works effectively for cases in which group information is drawn from sensor transmissions. The temporal correlations determine how much a sensor is deviating from its previous reading. We do not consider the case of a compromised sensor to be a faulty behavior. The sensor has enough storage capacity on it to hold values of both its neighbors' and its own until such a time the kth state is estimated. This comparison of values is used in the decision making process i.e. to send or discard sending a message.

B. EVENT DETECTION AND REPORTING ALGORITHM

Our approach uses a co-variance as a measure of the deviation the model should exhibit under the observed conditions. We expect both the detection of events and the subsequent reporting to be very accurate. Each, however, will depend on how fast the filter converges. The error covariance of a Kalman filter defines the amount of deviation from the actual value the estimated value can have. Thus, threshold required for an event to be detected is placed as the maximum covariance observed after the filter converges.

An event will be reported to the base station when a particular measurement crosses the threshold under a certain noise model. As previously mentioned, the predicted value is set as the reference for the correlated sensors' measurements. This value is first computed and then sent in an adaptive manner. This method of reporting is done to reduce the total number of messages that will be exchanged in the network.

Typically, the observed values that are observed should be exchanged at every time step for data fusion. This, however, is undesired in real-time scenarios, as events do not occur every time. In our approach, we update the base station only when an event is predicted based on previous observations. The sensor sends a message to the base station that an event could occur based on the current system model. The base station takes necessary precautions accordingly. These precautions are important as, in underground mines, monitoring sudden combustion is difficult. According to previously conducted studies on

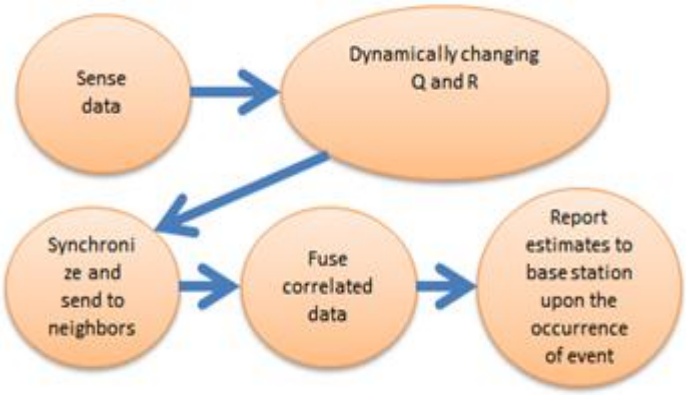

Figure 2. Graphical flow representing our scheme

underground mines, fire is reported at least five minutes after its onset. Our approach predicts this event before that time to allow workers the opportunity to take precautionary measures. When this prediction is a false negative, the filter in the sensor updates its input parameters.

The time at which the sensors calculate the estimates and communicate with other sensors is calculated in an adaptive manner. The filter learns about the behavior of the environment. It then dynamically changes the time instances at which data collection, estimation, prediction and communication are mandatory. A sensor may also request information from its neighbors if its observation shows an anomaly. The worst-case scenario, when the number of communications becomes maximum, occurs when all of the sensors exchange information at every time an observation is made. This scenario cannot be the case in practical applications. Thus, we can say that our approach reduces the total number of messages for the communication. Additionally, the sensors are updated dynamically with the noise parameters to give accurate results.

VII. EXPERIMENTAL EVALUATION

A. EXPERIMENTAL SETUP

An underground mine scenario was simulated using FLUENT. The conditions used to simulate a flow both in and out of a gob in a mineshaft have been considered [14]. They were implemented based on the C subroutines mentioned in [15]. A coal mine was considered as the test bed, and release of gases is from coal combustion. This release rate was determined by several parameters, such as atmospheric pressure and temperature, in the shaft at that time. For more information on this, [14] discusses the effects of atmospheric pressure on the flow of gases into and out from the mine gobs. Any sudden change in pressure might diffuse more gas into the mineshaft from the gob. This change could be misinterpreted as a possible fire hazard by the sensors monitoring the release of gases in the mine. Such a scenario was modeled using FLUENT to obtain a complete flow of gases in the mineshaft. In FLUENT, the deployment of the nodes is done as follows: the information for corresponding sensor nodes (assumed to be placed in several locations in the simulated mine shaft) is done by choosing what are called *nodes* in the environment generated.

The network setup is a distributed architecture. The nodes are located at different areas within the mine to monitor the release rates at the corresponding location. The nodes are connected to the base station in an efficient manner, with nodes in geographic vicinity being fully connected to one another. Hence, the spatial structure is built. The simulation was run over a 22-day period. An observation was taken every hour. Tossim was used to simulate the behavior on sensors. In accordance with the nodes' placement in the simulated mine, sensor network was simulated using TinyOS to record the communication among nodes. In order to obtain the execution times and power consumption, AVRORA

(a set of simulation and analysis tools for programs written for the AVR microcontroller produced for Mica2 and Atmel sensor motes) was used. These sensors were grouped into three groups of four sensors each. The approach we used to update the sensor system is a distributed approach. We simulated previously run algorithmic implementations on our test-bed to compare our scheme with existing schemes [5, 10]. The advantages of those schemes have been discussed in the related works section.

B. SIMULATION RESULTS

We ran our simulations over a 22-day time frame according to the simulation test bed. Both the inflow and outflow of gases from the gobs were the result of changes in the barometric pressure. The noise model of the system changed dynamically almost every day. The process noise was calculated according to the changes in the measurement noise. The covariance value of the measurement noise was varied between 0.75 and 2. The covariance value of the process noise was varied between 0.1 and 1. Initially to run the simulations, we considered the value of process noise covariance Q to be 1.5. The value of measurement noise covariance R was taken as 4 as the actual noise in the system was unknown. The new parameters are fed into the system and the fusion was completed after the simulation was initially run. Figure 3 illustrates the fused estimates at the time instants. When the values were not fused, the observed readings for the sensor implementing a lower R value were considered. That value was then sent to the base station. This value was chosen because, as R and Q approach 0 the observed and estimated values (respectively) become true.

We obtained the flow from the set of values that do not induce any anomaly. Thus, no events were detected in the gob. We introduced an artificial event to simulate the detection of the occurrence of an event. This was done by manually modifying the atmospheric pressure values to either increase or decrease the flow both into and out of the gob. As the barometric pressure decreased, the gob breathed out, making less oxygen available for coal oxidation.

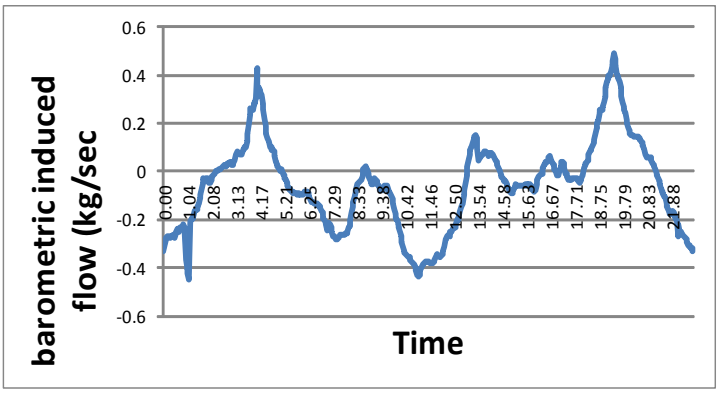

Figure 3. Barometric pressure induced flow

On the contrary, as the barometric pressure increased, the gob breathed in, making more oxygen available for coal oxidation. The maximum inflow was 0.46875 kg/s, and the maximum outflow was 0.43413 kg/s. These values were the actual values. The predicted values were 0.43455 kg/s for the inflow. The actual value before the manual change was 0.25506 kg/s.

Whenever the values went beyond the error covariance value of 0.2, the sensors reported the readings to the base station (see fig. 3). A spike occurs in the observed values at approximately day seven. This spike however (see figure 5), was not reported to the base station, as this is a false positive. The Kalman filter was then assessed for accuracy.

As we can see from the error covariance plot of our algorithm, the filter inputs were updated. The observed value was much lower than the estimated value at the time of actual observation. To determine the effectiveness of our scheme, we intentionally changed the values when prediction was complete. This indicates that our model works well when an event is detected. The error covariance varied when values of Q and R changed over a 22-day period (see figure 6).

The values converge soon enough to report an accurate value. A similar simulation

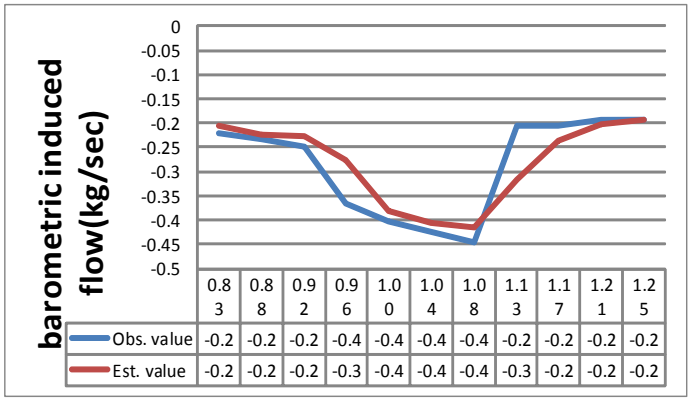

Figure 4. Blow up showing estimated values of the time when an event occurs

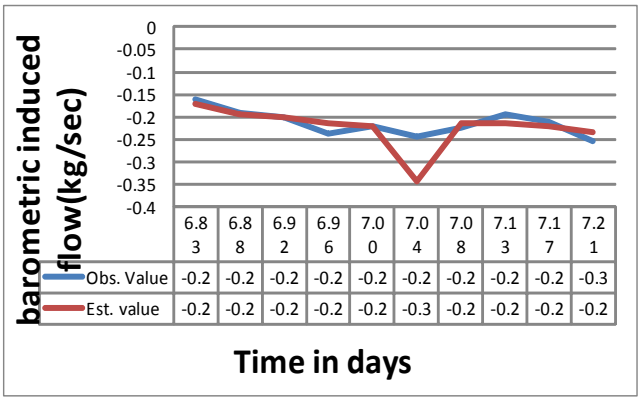

Figure 5. Illustration of False positive detection

was run for several runs, and all of the values concurred. In order to test the efficiency of our scheme we compared it with other schemes, Algorithm 1 [5] and Algorithm 2 [10]. The results of the error covariance show that our scheme clearly outperforms those schemes. Although in their schemes, the error covariance converged soon, and stabilized to a constant, this did not reflect the change in the noise. As we mentioned, this greatly affects the

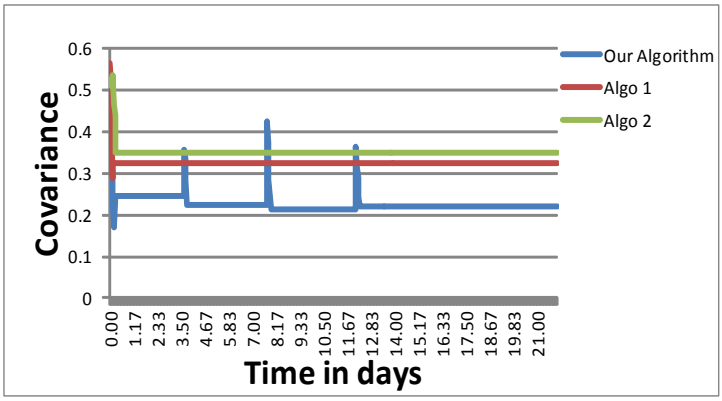

Figure 6. Error covariance at different times for the three algorithms

accuracy of the value estimated and reported after fusion.

C. NUMBER OF MESSAGES COMMUNICATED

The algorithm was implemented and the results were generated using TinyOS. The results of the flow through the gob concurred with those generated using MATLAB simulation. As we can see from figure 7, the number of messages generated initially was large. This is because, all the sensors exchange messages after sensing the data, estimating the corresponding states and computing the predicted state estimates. Knowledge obtained from all of the sensors in the group will enable the individual sensors to obtain global state knowledge. This helps in more accurately determining the local states. After initially converging to a reasonably accurate state estimation, the communication process takes place only when the sensors sense an event. This is when there is a gas inflow into the gob, or, when the covariance values go beyond the accepted threshold of 0.3.

D. TOTAL ENERGY CONSUMPTION

The energy computation was computed per-mote over a 24-hour period for all the

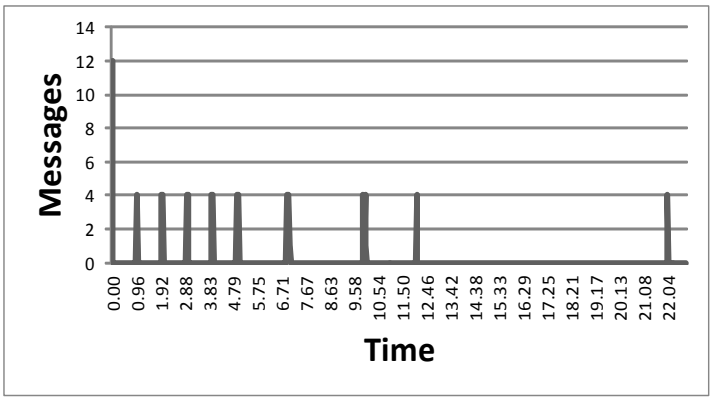

Figure 7. Total number of messages communicated

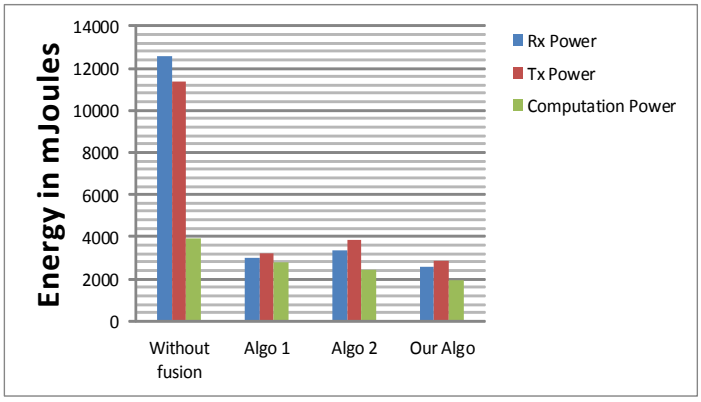

Figure 8. Energy consumption comparisons

three schemes for comparison purposes. We assume the sleep mode to be a state where the radio is still on and the sensor can listen to any messages. During this state, the computation was temporarily halted. As we can see from figure 8, the energy consumed in our algorithm was less than half when compared to an instance in which no fusion occurred. Our algorithm has an extra 381.6 mJ at the start of the fusion process.

This is because of the extra computational time of the initial estimates and for conver-

gence of the filter. This compensates for the amount of energy that is saved thereafter where computations were not required every time a senor sensed a new value. This, as compared to Algorithm 1 [5] and Algorithm 2 [10] was much less. Those schemes have a higher computational overhead as compared to ours. This is owing to the fact that we avoided using complex matrices. The computational cost for Algorithm 1 is higher than Algorithm 2 as they considered the entire network during fusion. Algorithm 2 uses a reduced matrix in its fusion computation. We can also see from the graph that the power consumption while receiving R_x and transmitting T_x is about less than a quarter as compared to the instance in which there was no fusion. This performance is better than those two algorithms in comparison. The sensors kept sending the messages to check for accuracy which incurred power consumption. This power consumption was maximum and largely only for the first day.

From the results obtained, we see that our algorithm works well in all the aspects of data fusion, viz. accurate state estimation in uncertain environmental conditions, reducing the number of messages thereby greatly reducing the computation and communication costs which are bound to extend the life-time of the battery.

E. APPLICATION IN UNDERGROUND MINES

In underground mines, ventilation systems are of utmost importance. This is because they regulate the flow of air in the mine shafts. Critical parameters for mine safety like temperature control, gas dilution, gas detection etc. require this regulation to be perfectly monitored. Existing procedures in mines do not make use of the sensors to actuate the process, thereby, requiring constant human monitoring. Raising any alerts (relating to fire etc.) to the central mine-safety system takes time. Prevention of mine fires has not been tackled effectively. Also, the detection is slow and hazardous. Under such scenarios, our algorithmic implementation allows us to actuate the mine ventilation system with the help of sensors. In essence, the algorithm acts as a feedback to the actuation system. The

sensors send the readings, fuse the values observed and then send them to the base station/ actuating system. The control messages from the actuators then regulate the flow of air in the mine shaft as per the requirement. The proposed algorithm can also be applied to a SCADA(Supervisory Control and Data Acquisition) system.

VIII. CONCLUSIONS

We proposed a method to estimate the unknown noise parameters in the sensor network. We developed an efficient method for calculating these values on power constrained sensor motes as the computations do not consider any matrices and inverses. In order to test our algorithm on a real-time environment, the simulation of the underground mine was done using FLUENT. The values input to the simulation were the ones we obtained from real-time readings in a mine. The simulation was done for several runs for randomly generated noise. We then ran the simulation in TinyOS to test the working of the algorithm on a sensor network scenario. Using the number of messages and the energy consumption, we conclude that the algorithm works well even in highly unstable system giving us the desired results. Our experimental results show that our model is fault resistant to a large extant. The results obtained thus proved the effectiveness of our fusion scheme. They were then compared with existing schemes to see where it stood. In future, we would like to extend our model for a multiple level data fusion process.

REFERENCES

[1] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," Transaction of the ASME-Journal of Basic Engineering, pp. 35-45, March 1960.

[2] X. R. Li, Y. M. Zhu, and C. Z. Han, "Unified Optimal Linear Estimation Fusion-Part I, II and III: Unified model and Fusion Rules," in Proceedings of the 2000 International Conference on Information Theory, 2000, paper MoC2-10-MoC2-17.

[3] C. J. Ran and Z. L. Deng, "Two average weighted measurement fusion Kalman filtering algorithms in sensor networks," in 7th World Congress on Intelligent Control and Automation (WCICA), 2008, pp. 2387-2391.

[4] Ribeiro, I. D. Schizas, S. I. Roumeliotis, and G. B. Giannakis, "Kalman Filtering in Wireless Sensor Networks: Incorporating Communication Cost in State Estimation Problems," IEEE Control Systems Magazine, April 2010.

[5] U. A. Khan and J. F. Moura, "Distributed Kalman Filters in Sensor Networks: Bipartite fusion graphs," in 14th Workshop on Statistical Signal Processing (SSP), 2007, p. 700 - 70.

[6] A. Kumar, M. Wang, L. Tong, and A. Swami, "Prize-Collecting Data Fusion for Cost-Performance Tradeoff in Distributed Inference," in Proc. of IEEE INFOCOM, 2009.

[7] J. Wang, Y. Liu, and S. K. Das, "Energy Efficient Data Gathering in Wireless Sensor Networks with Asynchronous Sampling," ACM Transactions on Sensor Networks, vol. 6, May 2010.

[8] R. Olfati-Saber, "Distributed Kalman Filtering for Sensor Networks," in Proceedings of the 46th Conference on Decision and Control, 2007, p. 5492-5498.

[9] S. L. Sun and Z. L. Deng, "Multi-sensor optimal information Fusion Kalman Filter," Automatica, vol. 40, pp.1017-1023, June 2004.

[10] Abdelgawad and M. Bayoumi, "Low-Power Distributed Kalman Filter for Wireless Sensor Networks," EURASIP Journal on Embedded Systems, vol. 2011, Article ID 693150, 11 pages, 2011. doi:10.1155/2011/693150.

[11] D. Simon, Optimal State Estimation: Kalman, H-infinity, and Nonlinear Approaches, 2nd edition, John Wiley & Sons, 2006.

[12] R. Tan, G. Xing, X. Liu, J. Yao, and Z. Yuan, "Adaptive Calibration for Fusion-based Wireless Sensor Networks," in IEEE INFOCOM, 2010.

[13] S. A. Aldosari and J. M. F. Moura, "Fusion in Sensor Networks with Communication Constraints," in Information Processing in Sensor Networks (IPSN'04), 2004.

[14] L. Yuan and A. C. Smith, "Modeling the Effect of Barometric Pressure Changes on Spontaneous Heating In Bleeder less Longwall Panels," NIOSH, 2010.

[15] Smith, A. C. & Yuan, L (2008). Simulation of spontaneous heating in longwall gob area with a bleederless ventilation system, SME Annual Meeting, Feb. 24-27, Salt Lake City, UT, Preprint 08-043

[16] G. Welch and G. Bishop (2001) An Introduction to the Kalman Filter [Online].

[17] Available: http://www.cs.unc.edu/welch/kalman/kalmanIntro.html.

[18] S. Ping, "Delay measurement time synchronization for wireless sensor networks", Intel Research, IRB-TR-03- 013, June 2003.

II. AN INFORMATION FUSION ARCHITECTURE FOR SECURE AND ROBUST WIRELESS NETOWRKED CONTROL SYSTEM

Brijesh Kashyap Chejerla, Sanjay K Madria

Department of Computer Science,

Missouri University of Science and Technology, Rolla, Missouri 65401

Wireless sensor network security essentially governs the usability and stability in control systems such as modeled by Networked Control Systems. We propose an information fusion architecture, which allows the profiling of different attacks on wireless sensor networks used in such applications and the study of their effects on the control system's stability. We use a Dynamic Bayesian Network to obtain control decisions which govern the control messages that are then input to the actuators. The decision making process allows us to ensure that the Wireless Network Controlled System (WNCS) works smoothly without any aberrations, or, at worst, has a graceful degradation even under the influence of security attacks. We consider the theoretical stability of a WNCS and the bounds on transmission delays in order to obtain a robust system. Using the proposed information fusion architecture, we are able to detect the presence of both stand-alone and collaborative attacks on the WNCS. New control messages sent to the actuators were transmitted based on these decisions which ensure system stability and security. Experiments were performed to validate our claim of providing the WNCS with robustness under security attacks. Our results showed that our fusion architecture can detect collaborative attacks and profile them with an average accuracy of 91.7%. Given the difficulty in detecting the presence of collaborative attacks in WNCS, this level of accuracy is high.

I. INTRODUCTION

Wireless Sensor Networks (WSNs) have numerous applications when data gathered by sensors is used to monitor and/or control the behavior of the system under observation. WSNs are also used in control systems such as power plant systems, home automation systems and in other Cyber-Physical Systems (CPS) like road transportation etc.

In an industrial setting, Networked Control Systems (NCSs) are Supervisory Control and Data Acquisition (SCADA) systems which comprise of a set of sensors to report readings/ observations and an actuator system which controls the plant operations. Traditionally, the readings sent by a set of sensors are processed at the control station. The control messages are then sent to the actuators. These actuators control the system operation as per the control unit's commands in real time to maintain its stability. The most important aspect of a control system is its operational stability as an unstable control system cannot be operational. Stability of a system is defined as the operational condition where even under errors or faults the system still yields finite and bounded outputs. System stability is governed by both functional faults and induced faults, where, induced faults could largely be a result of security breaches. Due to the time criticality of the information transmitted in a control system, security plays a major role in maintaining the system stability. In this paper, we attempt to mitigate system stability issues in NCS caused by security concerns.

Providing security mechanisms to any system is in itself a challenging task. In an application like Wireless Networked Control System (WNCS), where, the data is transmitted wirelessly, providing security becomes all the more challenging. An attacker can perform stand alone or collaborative attacks to disrupt system security, thereby, rendering the system unstable. Security attacks that compromise the NCS can be related to both integrity violation and data loss. Insofar, most of the existing works have focused on upholding the integrity of the data packets sent. In order to tackle this problem of security, many en-

cryption schemes [1, 2, 3, 1, 5] have been employed in WSNs to protect both privacy and integrity of the data. Although data is encrypted, encryption alone is not enough to provide comprehensive security to a control system.

Scheduling is another crucial aspect that plays a very important role in maintaining stability of a WNCS. Hence, its vulnerability towards attacks is high. Several attack scenarios exist with both a single attacker and multiple attackers. A stand-alone attacker may launch several attacks or multiple attackers may collaborate and launch multiple attacks. Many of these attacks look to exploit the vulnerabilities of the existing solutions to disrupt the scheduling and reporting process. Hence, care must be taken to develop a method that detects the presence of an attacker while simultaneously thwarting the attacker from further disrupting the stability of the plant. To tackle this problem, we propose an information fusion architecture which co-designs control and automation and allows the WNCS to work effectively even under the influence of attacks (both collaborative and stand alone).

In WNCS, the network is shared by controller nodes and actuator nodes. Therefore, it is extremely crucial to ensure appropriate scheduling among the nodes to account for transmission times that uphold stability. Scheduling the nodes to transmit information in both open loop and closed loop conditions is a major research area. In addition to the scheduling problem, exogenous disturbances into the system cause further complications that have to be addressed. Exogenous disturbances could exist because of a malicious attacker, in which case it is difficult to control the input of such disturbances. This warrants an architecture that detects the presence of an attacker and mitigates the losses caused by his/her activities. Our information fusion architecture works towards this goal by studying the different conditions that cause the WNCS to become unstable and provides solutions that keep the WNCS stable.

To achieve the goal of stability, researchers have focused primarily on control engineering where disturbances input to the system can be known. Contrary to this research direction, our focus is to provide operational stability by providing comprehensive security

framework against collaborative attacks. Our security framework ensures the functional robustness of the WNCS by providing graceful degradation in situations where most security mechanism fail.

To this effect, we built an intelligent information fusion architecture on top of the WNCS. It acts as an automation system which chooses from among a set of decisions and takes required actions. These actions are based on the hypotheses generated according to the outputs of a dynamic Bayesian Network (DBN). Conditional probabilities are used to define the causal relationships among the nodes in the DBN. We provision the handling of non-Gaussian distributions in our framework, thereby, increasing the solution scope beyond the limitations of Gaussian distributed data. The ability to provide an integrated solution to profile different attacks (both stand alone and collaborative) is the defining feature of our approach. We were able to monitor the system for attacks while simultaneously providing stability thereby robustness. To the best of our knowledge, no prior work in this area has provided an integrated fusion architecture used in co-designing stability and security of a WNCS under the influence of security attacks.

To summarize, the primary contribution of this paper is as follows:

(i) Study the stability constraints of a WNCS under long delays, quantization effects and sampling delays

(ii) Propose a novel time-varying dynamic Bayesian network (TVDBN) aided by feature extraction

(iii) Effectively profile both stand-alone and collaborative attacks to accurately represent network behavior

(iv) Enable accurate decision making under security attacks by proposing an Information Fusion Architecture

The rest of the paper is organized as follows: We formally define the problem in section II. Section III gives an understanding of theoretical constraints on transmission

times and delays, time varying sampling and quantization in maintaining system stability,. Section B gives an in-depth understanding of our implementation architecture. We then discuss the simulation results in A and analyze them. Section B talks in detail about the advantages and drawbacks of existing works in the field. Finally, we conclude the paper with our remarks about our contribution in section B.

II. PROBLEM STATEMENT

The problem studied is that of providing a security framework which guarantees the stability and operability of a WNCS under security attacks. Security attacks can be profiled as stand alone or collaborative attacks. We address the problem of security to support the theoretical stability constraints of a WNCS.

Let us consider a continuous time plant model

$$\dot{x_p} = f_p(x_p, \hat{u}, w), \quad y = g_p(x_p) \tag{1}$$

where, f_p and g_p are continuous time functions and x_p, $\hat{u}$, w are the state input and noise/disturbance variables respectively and y is the output of the plant. At the time instant t_{s_i} (sampling time), the inputs to the actuator or the observations from the sensor node are transmitted. The transmission/sampling times follow $0 \leq t_{s_0} < t_{s_1} < t_{s_2} < t_{s_3} \ldots$. and $\exists$ a $\delta > 0$ such that the transmission intervals given by $t_{s_i} - t_s$ are bounded by $\delta < t_{s_i} - t_{s_{i-1}} < \tau_{mati} \forall i \in N$, where, τ_{mati} is the Maximally Allowable Transmission Interval (MATI). For every sampling time instant t_{s_i} a set of nodes $n_j \in N_j$, $j \in 1, 2, 3, \ldots$ is chosen for data transmission. The information collected from the sensors are $y(t_{s_i})$ and $\hat{u}(t_{s_i})$. The information sent over the network has a delay of τ_d. This delay encompasses $\tau_{communication}$ and $\tau_{processing}$. In other words, the plant gets back the control signals after $t_{s_i} + \tau_d$ time units.

Under exogenous disturbances, maximally allowable delay $\tau_{mad} = \max\{\tau_d\}$ can grow larger than τ_{mati}. This is a hard problem to solve. This delay can be caused by an attacker who has control over a portion of the network. Hence, it is of utmost priority to detect the presence and the effect of an attacker in the WNCS. In addition to delays over the communication channel, the attacker can cause delays by disrupting the scheduling function of

the WNCS. There are multiple methods in which he can do this. Some of them are varying sampling times and ineffective quantization. For instance, reducing quantization errors results in the transmission of larger packets which could affect the delays (τ_{mad}). It must be noted that quantization is not a serious problem in packets sent over internet traffic, but is a considerable problem in WNCS where sensors are used with a limit on the packet size. An attacker can also affect the sampling times and cause both energy depletion of wireless nodes and unwanted rescheduling of the network. This could cause some of the nodes to be discarded from the network, in which case the attacker can continue his malpractices.

To this effect there is a very serious need to study the most pertinent attacks that can be employed to cause system instability and inoperability, and develop an architecture that detects and avoids such attacks. Hence in this paper, we propose an Information Fusion Architecture.

In the next section, we go on to describe the stability considerations needed to provide security.

III. STABILITY ANALYSIS AND TRANSMISSION TIME BOUNDS OF WNCS

A. DELAYS DUE TO PACKET DROPS

Let us again, consider the continuous time plant model of an NCS as eq. (1). There is a bound on the total delay that can be allowed for the system to be stable. Delay $\tau_d \leq \tau_{mad}$ where, τ_{mad} is the Maximally Allowable Delay (MAD). To put both τ_{mati} and τ_{mad} into perspective, we can state the time bounds as follows:

$$\delta \leq t_{s_{i+1}} - t_{s_i} \leq \tau_{mati}, i \in N \tag{2}$$

$$0 \leq \tau_d \leq \min\{\tau_{mad}, t_{s_{i+1}} - t_{s_i}\}, i \in N \tag{3}$$

The above two conditions imply that the data is transmitted after it is sampled and arrives only before the next sample is taken. These conditions will be violated in the case of a transmission delay larger than τ_{mati}. Incorporating cases where the delays are larger than the maximally allowable time interval is a hard problem and requires intelligent decisions to overcome it. Although, use of bounds on transmission times can be used directly to determine the DoS and Jamming attacks, they are not wholly sufficient in determining the nature of the attack. One must have additional information regarding the packet drops etc. Assume that the number of successive packet drops is η, then with each successful successive packet drop, the new MATI becomes $\tau_{mati}^{'} = \tau_{mati}/\eta + 1$. In other words, the maximally allowable time interval decreases.

In order to complete the stability analysis, we have to consider the error in the system which is governed by the function that determines it at every time instant. The amount of error that is allowed to creep in is based on the nodes that are chosen to transmit data over the network. Now, let the observed and input values after $t_{s_i} + \tau_d$ be

$$\hat{y}(t_{s_i}+\tau_d) = y(t_{s_i}) + \dot{f}(i,\ \dot{\varepsilon}\ (t_{s_i})) \tag{4}$$

$$\hat{u}(t_{s_i}+\tau_d) = u(t_{s_i}) + \dot{f}(i,\ \dot{\varepsilon}\ (t_{s_i})) \tag{5}$$

where, $\dot{\varepsilon}(t_{s_i}) = \hat{y_{t_{s_i}}} - \hat{y_{t_{s_{i-1}}}}$; $\hat{u_{t_{s_i}}} - \hat{u_{t_{s_{i-1}}}}$ is the error and $\dot{f}$ is a function that determines the nodes that are chosen at the time instant i.

B. STABILITY ANALYSIS BASED ON DIFFERENT CONSTRAINTS

The stability analysis of a NCS and thereby a WNCS is based on [6]. Let $\tilde{W}(\kappa,\ell,\dot{\varepsilon},s)$ be a Lyapunov function satisfying

$$\tilde{W}(\kappa+1,1,\dot{\varepsilon},\dot{f}(\kappa,\dot{\varepsilon})-\dot{\varepsilon}) \leq \lambda\tilde{W}(\kappa,0,\dot{\varepsilon},s) \tag{6}$$

$$\tilde{W}(\kappa,0,s+\dot{\varepsilon},-s-\dot{\varepsilon}) \leq \tilde{W}(\kappa,1,\dot{\varepsilon},s) \tag{7}$$

$\forall\kappa \in N,\ \ell \in \{0,1\}$ and $\dot{\varepsilon},s \in \mathbb{R}^{n_e}; n_e = n_y, n_u$, where, specifically, ℓ is a boolean which determines if there is a data or control packet transmission, or if there is a system state update; ε is the error and s is the arbitrary variable that holds the value corresponding to $\dot{f}(i,\dot{\varepsilon}\ (t_{s_i}))$ and κ is a counter that keeps track of the number of transmissions and $0\ \leq\ \lambda\ <\ 1$.

Consider the following differential equations

$$\dot{\phi_0} = -2L_0\phi_0 - \gamma_0(\phi_0^2+1) \tag{8}$$

$$\dot{\phi_1} = -2L_0\phi_1 - \gamma_0(\phi_1^2+\frac{\gamma_1^2}{\gamma_0^2})$$

These equations describe the behavior of the plant based on the initial conditions ϕ_0

and ϕ_1. Here, $L_\ell \geq 0$ and $\gamma_\ell > 0$, $\ell = 0,1$ are constants of the locally Lipschitz function $\tilde{W}(\kappa, \ell, \dot{\varepsilon}, s)$.

The following theorem gives the stability of an NCS conditioned upon τ_{mati} and τ_{mad}.

Theorem 1. Consider an NCS that satisfies the condition that there exits a Lyapunov function $\tilde{W}(\kappa, \ell, \dot{\varepsilon}, s)$ which is locally Lipschitz. Now, if $\tau_{mati} \geq 0$ and $\tau_{mad} \geq 0$ satisfy

$$\phi_0(\tau_i) \geq \lambda^2 \phi_1(0) \;\; \forall\, 0 \leq \tau_i \leq \tau_{mati} \tag{9}$$

$$\phi_1(\tau_i) \geq \phi_0(\tau_i) \;\; \forall\, 0 \leq \tau_i \leq \tau_{mad} \tag{10}$$

for various solutions of ϕ_0 and ϕ_1 from (8), with corresponding initial conditions $\phi_\ell > 0$, $\ell = 0,1$ and $\phi_1(0) \geq \phi_0(0) \geq \lambda^2 \phi_1(0) \geq 0$ with $0 \leq \lambda < 1$, then the system is considered to be Uniform Globally Asymptotically Stable (UGAS).

The proof of this theorem is given in [6] and its Lyapunov stability proof is discussed. From the proof, the system is both UGAS and Uniformly Globally Exponentially Stable (UGES) based on Lyapunov arguments. For the case of Input to State Stability (ISS), $\mathcal{L}_p$ stability proof is also provided under delay considerations. Our contribution is the inclusion of long delay ($\tau_{mad} > \tau_{mati}$) case that cause system instability which is not discussed in [6]. In order to incorporate long delays, we alter the condition in (9) and (10), i.e. we assume that the reception and transmission times for the actuator and sensor nodes are asymmetric. In order to solve this problem, we propose to use a longer control packet without adding any further delays, which holds the value of the future control messages based on the output generated up to that time instant. In our implementation, since the control messages are sent based on the determination of the τ_{mati}, if $\tau_{mad} > \tau_{mati}$ is observed, the estimated control input $\hat{u}_{t_{s_i}}$ is fed into the actuator which becomes the latest control input to the plant before τ_{mati} expires. This does not violate the stability proof at the same time solves the problem of lost or delayed control packets to the actuator. The decision of sending larger control packet is taken based on the hypotheses that is generated from our fusion architecture.

From the theorem the quantitative numbers for τ_{mati} and τ_{mad} are obtained by constructing solutions to (8). In order to determine τ_{mati}, the intersection of ϕ_0 and the constant line $\lambda^2\phi_1(0)$ is observed and to obtain τ_{mad}, the intersection of ϕ_0 and ϕ_1 is observed. Different values of the initial conditions lead to different $\tau = \{\tau_{mati}, \tau_{mad}\}$ values. Hence, based on the different initial conditions as observed by our information fusion architecture, we adjust the τ values accordingly to obtain system stability. Its study is provided in section A.

A. Time-Varying Sampling times.

In addition to the stability constraints given above for the τ_{mati}, we now consider the stability under time varying sampling times. It is considered since constant sampling in WNCS cannot be assumed. Consider the existence of Lyapunov functions as described above, the closed loop system is semi-globally practically stable if eq. (8) is parametrized by a modeling parameter h which is in turn based on the sampling time. Decision about the modeling parameter is based on the input conditions of the WNCS and network parameters such as bandwidth, sensor data rate etc. At each time interval, h is a constant and $h > \tau_{mad} \geq 0$, then for the discrete-time representation of (1), $t_{s_{i+1}} - t_{s_i} = h$, the WNCS is exponentially stable iff $\Phi(h, \tau_{mad})$ has eigen values that are strictly less than one and $\Phi(\dot{})$ is a exponential function.

Now, let $C_{seq} = \{0 \leq t_{s_0} < t_{s_1} < t_{s_2} < t_{s_3} ....\}$ be a communication sequence for the plant. For each sampling time t_{s_i}, the plant state $x(i)$ is measured and the corresponding data are sent over the network to the controller node. Also, for each t_{s_i}, let τ_{RTT} be the corresponding total round-trip time (RTT) delay. This means that, if a state measurement is performed at i^{th} sampling instant, t_{s_i}, the corresponding control action is available at the actuator starting from $t_{s_i} + \tau_{RTT}$ sampling instant. At this point, no restrictions are applied on C_{seq} and τ_{RTT}; in particular, it is possible that $\tau_{RTT} = +\infty$ for some $t_{s_i} \in C_{seq}$, which corresponds to the case of packet dropout. Again as in the previous section based on the initial conditions $\phi_i, i \in \{1, 2\}$ and the values of τ_{mati} thus determined, we can determine the upper bound on τ_{RTT}. Based on this bound, the sampling rates and sampling times have

to be changed.

B. Quantization.

The impact of quantization in the system will lead to controlled output of behaviors such as limit circle, dead zones and chaos. This, in addition to security loop holes can create more disturbance in the system than the system can handle and it is important to ensure stability under the constraints of quantization. In a wired network control system, the effect of quantization does not really account for much disturbance owing to the presence of large packet sizes. However, in a wireless setting, where the control plants analog signals are sampled periodically, quantization plays a major role in determining the overall system stability and accuracy. It is important to ensure a bounded-input bounded-output stability in this case as Global Asymptotic Stability does not talk about the cases when the convergence point is not at or around the origin. In other words, WNCS will be stable for any bounded input, which means it will always yield a bounded output. For the same plant, a quantizer is a piecewise constant function $q : R^n \rightarrow Q$,where Q is a finite subset of real numbers R^n and n is number of nodes. We assume that there exist real numbers such that the following two conditions hold:

$$|z| \leq M \Rightarrow |q(z) - z| \leq \Delta \tag{11}$$

and,

$$|z| > M \Rightarrow |q(z) - z| > M - \Delta \tag{12}$$

where, M and Δ are the range and error bound of the quantizer, and z is the quantized value respectively. The first condition gives a bound on the quantization error when the quantizer does not saturate, while the second one provides a way to detect the possibility of

saturation. Based on these conditions, the bound on the acceptable quantization errors can be calculated depending on z which is the quantized value of the state x. In order to avoid the quantization problem, we propose to alter the sampling frequency as required. Also, the error in quantization will be useful in the determination of allowable residual value (ref C). We refer the readers to [7] for a more technical and detailed study of quantization induced delays.

IV. FUSION ARCHITECTURE

A. OVERVIEW

The proposed information fusion scheme takes into account the previously mentioned parameters, and outputs control messages that both stabilize and secure the WNCS. We used a two layer approach to tackle both detection and eventual prevention of attacks.

Level 1 comprises of extracting features from the incoming stream of data and the traffic patterns. Based on this feature extraction, a causal graph structure is generated with nodes representing the different parameters. Causality is determined by the correlation among parameters and the features of the network. These features include the attacks and the network behavior. In order to capture the dynamic nature of the sensor network, we then made use of a Dynamic Bayesian Network (DBN). The DBN unrolls itself over time and the next state is affected only by the previous state. The hidden state gives us an understanding of the system transition before outputting the final state. The input to the DBN were the probability values of

1. the occurrence of the parameters
2. conditional occurrence of the features of the network

With the continuous stream of observed data being fed from the sensors, DBN is unrolled over continuous time slices and the attacks are profiled and differentiated from among other existing and known attacks using feature extraction. Initial probability values were input to the DBN based on the probability of occurrence of an attack based on existing literature. As our model uses a DBN, the probability values are updated over time slices and eventually, the BN converges to give an accurate representation of the system behavior.

Feature extraction is done based on the temporal information obtained from the NCS data traffic. Information such as the second order moments of the traffic trace and temporal

auto-correlation among time slices yields the most appropriate feature of the WNCS. Unrolling the causal graph over time slices unless done without including system unknowns will result in faulty WNCS analysis. In our architecture, system unknown is defined as a probability value which reflects the lack of absolute knowledge of the system's parameters. Feature extraction is of utmost importance as the content of the control signal input to the actuators is determined by its accuracy.

Level 2 comprises of further pruning the edges of the causal graph based on feature extraction, using DBN for decision making for further information acquisition and hypotheses selection, formulating the content of control packets and altering the scheduling strategies and data rates of the sensors to suit the needs of the WNCS.

B. FEATURE EXTRACTION

A. Level 1.

Firstly, the causal network is constructed with arcs towards all the nodes (parameters) that exhibit causality among each other. Later, this causal graph transforms into a Bayesian network. The strength of the causality (arc weights) is determined by factors such as correlation among the nodes and conditional probabilities (which determine the dependency among parameters in selecting a feature of the network). The features of the network that are chosen are the attacks on the system. The parameters that are chosen are: *packet drops, inconsistent data, end to end delays, communication delays, system delays, propagation delays, inter-arrival times, insider and outsider attacker, quantization errors and sampling time* $t_{sampling}$. The attacks that are profiled based on these parameters include, but not limited to, stealthy attacks - data deception attack and replay attacks, and Denial of Service (DoS) attacks (e.g., Wormhole attacks, Energy drain attacks, and jamming attacks).

In order to train the graph structure and to establish relationships among the nodes of the graph, we used the data set of a water plant provided in [8]. We conducted experiments using the data set by injecting the four attacks used in this paper into the data trace. This

was done in order to establish a difference in how the incoming data will be processed by our fusion algorithm upon simulating the plants behavior. The packet arrival rates, delay times and state estimate errors of the dataset under attacks were then compared with the introduction of different noise levels in the system but under no attack. It gave us an insight into how the network behaved under various conditions and helped us better determine the causality among arcs. We then went on to compute the inter-arrival times τ_{iat}, end-to-end delays τ_{e2e}, system delays τ_{sys}, successive packet drops η, error ε, sampling times τ_{sam}, and the quantization error ε_q which is determined by the initial conditions ϕ_i, $i \in \{1,2\}$. upon obtaining the values of these parameters, we looked at how the causal graph built itself and how the strength of the arcs based on correlations among nodes affected the feature extraction process.

The next step in Level 1 is graph pruning. In order to prune the graph and to obtain a locally optimal network structure, the graph has to be dynamically updated. This pruning was done to eliminate the edges which are less causally related and as a result do not affect the feature selection process. As there were many sensors to choose from, data fusion/aggregation was an obvious choice in order to eliminate redundancy and save the channel bandwidth. Use of AES was not considered as it does not support aggregation. However, it must be noted that even upon the use of a secure data aggregation scheme, the system will still be vulnerable to 'insider' attacks. Upon use of a secure data aggregation scheme in unison with our IFA, we can now call our scheme as SIFA (Secure Information Fusion Architecture).

Once the basic algorithm of Level 1 is run, the values of τ_{mati} and τ_{mad} can be found out for different input values of the differential equation (8). These values are then updated into the system and act as the bounds on the maximally allowable time interval and maximally allowable delay. In situations where the delay is larger than the time interval, the NCS scheduling, data collection and sampling is adjusted accordingly in Level 2. This is usually a hard problem to solve and our decision making system overcomes it as mentioned

in the discussion in sec. III. An algorithmic representation of Level 1 is provided in algorithm 2. During the initial run, Level 2 is FALSE, but in the subsequent unrolling of the DBN, control inputs from Level 2 are input to Level 1 and the process continues.

Algorithm 2: LEVEL 1 Feature Extraction

Input: Network parameters, Causal Graph G
Output: Features, Attacker nature, New Graph $\acute{G}$
if $Level2 == TRUE$ **then**
  $G \leftarrow \acute{G}$
  $control\ input\ \{u \leftarrow \acute{u}\}, \{\tau_{mati}, \tau_{mad} \leftarrow \tau_{\acute{m}ati}, \tau_{\acute{m}ad}\}$
else
  $\forall\ sensors\ s_i\ \in \{S\}, collect\ observations\ O_i,$
  $i \in \{1...n\}$
  while $system\ stability\ == TRUE$ **do**
    compute $\tau_{iat}, \tau_{e2e}, \tau_{sys}, \eta, error\ \varepsilon,\ \tau_{sam},\ \varepsilon_q$
    generate $PDF\ and\ Correlation\ R_{s_i,s_j} \forall s_i, s_j \in \{S\}$
    Perform SDA
    $Send\ values\ to\ Controller\ node$
    Update G
  end
  choose $new\ nodes\ to\ obtain\ information$
  return $Attack,\ nature\ of\ attacker, \tau_{mati}, \tau_{mad}$
end

C. SECURITY ATTACKS AND THEIR MITIGATION METHODS

A few of the attacks previously mentioned that are most pertinent to WSNs deployed in an industrial controls scenario have been studied. We give a brief description of them and how we intend to tackle them.

Denial of Service attack: This attack is the most common choice of attack to disrupt the control system operations. A Denial of Service (DoS) attack in a sensor network targets the resources of a sensor node thereby making them inactive after some time. This kind of

attack falls under the category of both an insider and an outsider attack. Message rerouting attacks, spoofing attacks and jamming attacks are some examples of DoS attacks. For a DoS attack, the probability that an attack is taking place is calculated by looking at the packet arrival rates, end-to-end delays and communication delays. Anomalies in the transmission times caused by genuine system noise are also considered to detect its presence. In this paper we consider the jamming attack. Wormhole attack is another DoS attack. Our objective is to be able to differentiate between the different types of attacks to better address the security issues. One of the methods (at the application layer) in which DoS attack can be countered is the data aggregation by forming secure clusters. We intend to mitigate the Jamming attack by flagging the nodes where packet drops and network delays are very frequent. Rerouting data through other sensors around such nodes and passing control messages that contain information about which sensors get to use the channel for data transmission helps mitigate the Jamming attack. Another commonly used approach is the usage of multiple frequency channels.

Wormhole attacks: The wormhole attack is one in which a sensor node receives a message at its origin and sends it to its destination. A wormhole attack is difficult to detect when packet delivery ratio is low owing to packet drops caused by system noise. A genuine packet drop due to poor network conditions can be mistaken as a wormhole attack. Under high noise levels, we obtained the packet arrival rates PDFs with different network parameters at the start of the network deployment. These PDFs were the reference PDFs with which the packet arrival rates of the sensors during real time was compared. We examined the total packet delivery ratio from different sensors (aggregator nodes and forwarding nodes). Doing so provided a better understanding of how long a sensor node had held a packet and where it sent the packet. The probability of an intruder in the system was derived according to these results. This probability aided in developing the overall probability of the presence of a standalone Wormhole attack or a collaborative attack. In a WSN environment, where the nodes are static, the solution for wormhole attack talk about

using multiple base stations. Rerouting traffic to other nodes is the most common solution. However, in a WNCS, this rerouting is not as straightforward as it seems. This is because, rerouting may affect the network scheduling and also, under collaborative attacks, rerouting may only aid an attacker more.

Stealthy attacks: Deception attacks in a sensor networks work by manipulating the data that is transmitted or by changing the data to be sent in a manner that allows corruption of data to go undetected. This kind of attack is difficult to detect. But for this attack to be successful the requirements grow manifold. Both an outsider and an insider can perform this attack. Over time, the probability that the data is inconsistent is calculated. This probability is fed as one of the conditional probabilities of the system. The different types of stealthy attacks are replay attacks, injection attacks, bias-injection attacks, data deception attacks etc. Below, we mention the two kinds of attacks that are implemented in this paper and our methods to detect them.

a) Data Deception attack: The system state that we studied can be written as

$$\dot{x}(t) = Ax(t) + Bu(t)y(t) = Cx(t) + Du(t)$$

where, $A \in R^{n\times n}$ is the input matrix, $C \in R^{k\times n}$ is the output matrix. Both $B \in R^{n\times m}$ and $D \in R^{k\times m}$ are attack matrices. These values define the attack state affecting the system input and the output attack matrix affecting the measurement vector respectively. For an observer,

$$\dot{z}(t) = Fz(t) + C_b u(t) + Ky(t)\hat{x}(t) = \dot{z}(t) + Hy(t)$$

where, z is the measurement (observed), $\hat{x}$ is the new estimate according to the observed measurement. H is the observation matrix for the original observed value y, F is the observation matrix and C_b is the observed attack matrix with the corrupted input. For a global observer, we have

$$\dot{x} = Ax(t) + Bu(t) + B_f a_t \dot{y}_i = J_i x(t) + Du(t)$$

where, $a_t = a1, a2, a3, ...a_n$ is the attack vector, J is the observation matrix corresponding to the estimate x and B_f should have a full column rank. The corrupted estimate is calculated as

$$z^k = F^k + T^k Bu(t) + K^k y(t) \hat{x}^k(t) = z^k + H^k y(t)$$

The estimation error $\dot{e} = Fe$ where, $e = \frac{\hat{x}}{z} - Hx$ is used to detect the attack. This error must be in a tolerable bound. Note that this estimation error can arise because of quantization errors which in turn arise from lower sampling rates. Lower sampling rates in WSNs are required to conserve energy on the sensor node. Here, F governs the stability of the system so that the error has to become zero over time. The measurement residual $\rho = z - H\hat{x}$, where z is a function of the input u and the actual measurement, and, H is the observation matrix, is the acceptable deviation in the value to maintain stability. The residual $\rho = H(\hat{x} - x)\dot{e}$ has to be less than a predetermined threshold value as governed by the Level 2 of the Information Fusion Architecture. This threshold value is determined based on its effectiveness in detecting the attack. It must be noted that a small residual will lead to discarding genuine data under relatively high noise, and, a higher residual will lead to faulty packets being accepted. Hence, determining the threshold value is of utmost importance in detecting this attack. Our information fusion architecture ensures that a locally optimum threshold value is chosen to best represent the system behavior i.e. determine the cause as either noise or attack. If an attack is detected this threshold is updated and it is minimized to reduce the effect of this type of attack.

b) Replay attack: For a replay attack considering the same system as (13); in order to detect the attack, we injected an unknown signal into the system at a random point in the system operation or at a chosen time. This signal is unknown to the attacker in the sense that system behavior with this input cannot be determined by the attacker to converge the state estimates to fit the expected pdf. The information inferred from the abnormality in system state is represented in the control input to the actuator according to the output of the

hypotheses selected. Therefore, we can detect the presence of an attacker. This is from the measurement attack vector that can be found out by

$$Du(t) = y(t) - C\hat{x}(t) \tag{13}$$

where, $\hat{x}(t)$ is the reconstructed state estimate from the new signal at the time of inducing the signal that the user injects into the system and is input at the corresponding time. Again if the value of $Du(t)$ is not within the accepted bounds, we confirm the presence of an attacker. In attacks where the attacker makes the difference between the attack state estimate and the actual state estimate $\hat{x} \sim x$ very low, there is a possibility that the attackers presence may go undetected. In such instances, however, the attack itself causes no harm to the system's stability. Furthermore, the attacker must vastly increase his attack scope (number of nodes to be attacked) to either create disruptions or steal any resources. We did not investigate an effective solution to this effect as gaining access to a large pool of sensor nodes is in itself a challenging task even to an attacker. As in the case of data deception attacks, the data sampling is altered according to the control information sent to the sensors and actuators from Level 2.

Based on the parameters and the procedure mentioned previously, feature extraction of several attack scenarios is done repetitively in order to reflect the most accurate representation of the WNCS. Information form the feature extraction is used to perform likelihood ratio tests for different test samples to help profile the various attacks. These samples are obtained from the control system's operation under attacks. Multiple hypotheses were generated and a few with the highest likelihood ratio were chosen according to the operation of the system under the various attack scenarios. This profiling and Hypotheses generation and choice is done offline before the information fusion system is integrated into the sensor network structure.

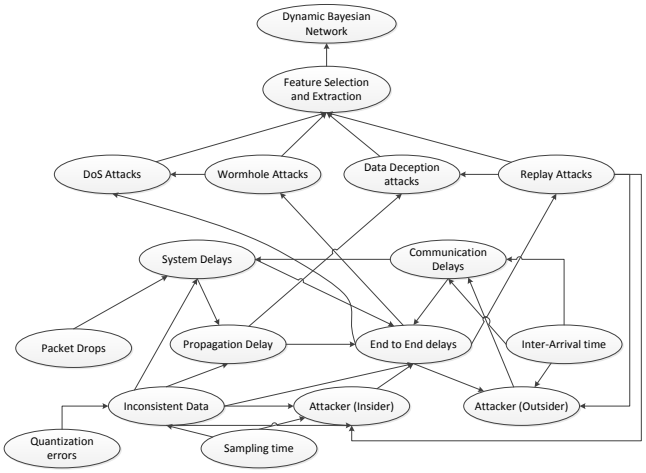

Figure 1. Bayesian Network Model for feature extraction and decision process

D. INFERENCE ALGORITHM

A. Level 2.

It consists of the time varying Dynamic Bayesian Network (TVDBN), the hypotheses selector and the control unit (whose output is fed into the actuators). The causal relationships in the Bayesian Network (BN) are initially generated from *a-priori* probabilities. These probabilities are derived from the information obtained from the feature extraction process. Once the causal relationships are built as graph G, information from the feature extraction process is fed periodically into the TVDBN. The TVDBN then unrolls over time slices and is conditioned upon only the input at the previous time slice. Edges are then either pruned or added according to the strength of probability values combined with the correlation among the nodes. This unrolling over time slices constitutes the dynamic aspect of the Bayesian network. In a real-time environment, the BN does not update very frequently. The provision to both dynamically adapt and update will prove helpful in the case of an attack.

In this paper, we use a novel time varying dynamic Bayesian network (TVDBN) model for online inferencing. We extend the basic DBN model so that both the structure G and parameter Θ of network such as *attacks, delays, quantization, packet arrival times etc.* become random variables that can change through time. These random variables $G[t]$ and $\Theta[t]$ are treated as additional hidden nodes in our graph model because they cannot be observed directly. Contrary to most off-line learning methods, we employ our feature extraction mechanism to dynamically infer the hidden states of network as well as missing data. Our framework is proposed in such a manner that any distribution, i.e. either multinomial or Gaussian is dealt with effectively. A smooth transition prior is imposed on their temporal variation to ease the problem of data scarcity. This novel representation of changing network allows a unified modeling of both data and network itself under the same dynamic Bayesian framework. In order to solve the problem of limited information to accurately obtain the conditional probabilities, we employ a feature extraction and selection method such as the particle filter. A particle filter (or Sequential Monte Carlo) is used to determine the state posteriori because it can handle arbitrary system and observation models. The posterior distribution is approximated by a finite set of state samples $\{s_t^i\}$ and its associated weights $\{w_t^i\}$.

$$p(s_t|o_{1:t} \approx \sum_{i=1}^{N_s} w_t^i \delta(s_t - s_t^i) \tag{14}$$

and when N_s approaches infinity, the approximation can be close to the true distribution. At each time epoch, filtering in done with the sample $\{s_{t-1}^i, w_{t-1}^i\}$ of the previous time. New samples are drawn from a proposal distribution $q(\bullet)$. With this, inference of the hidden states can be performed for different network structures and parameters with different dimensions.

We would like to mention the notable differences in our model and the ones that are

comparable to it. A hidden variable is used to represent the change of network in [20], but it only serves as an auxiliary variable to facilitate implementation. The structure and parameter nodes in our TVDBN are principal components of the whole model, and they represent the statistical attributes of current network. The online adaptation methods in [13] can only model piecewise constant variation of network; while our method deals with both continuous change in parameter and discrete change in structure. Smooth change of network is ensured in [17] with a kernel window applied on data sequences; we achieve similar goal via a smooth transition model with the aid of feature extraction and parameter selection for the generation of conditional probability, which is a more favorable solution from a Bayesian perspective. In the Gaussian graphical model [23], network parameter is marginalized out and network structure is the only thing that is investigated. This does not yield complete and accurate results as it cannot comprehensively determinate the causal relations. In our approach, the states of both structure G and parameter Θ are inferred to give a full description of current network.

As we deal with probability values, we can also use predictive mechanisms to determine the effect of one parameter on the entire network. This understanding allows us to generate hypotheses in the second phase of level 2 which comprises of hypotheses generation and testing. The generation is done according to the knowledge of the network behavior. The hypothesis that is chosen is then sent to the Bayes output generator. This generator then computes the best possible data rates, sampling, re-scheduling information for optimal performance and stability. The cause and effect of parameters chosen for the next time slice is determined and the control commands to the actuators are preset accordingly. The commands will eventually be updated according to network's performance and behavior. This entire process is possible because of the information acquisition and fusion. Our method also incorporates the possibility of an non-Gaussian input to the BN. We employ the Gaussian Mixture Model, which is a probability density function represented as

the weighted sum of the Gaussian component densities. It is given by

$$p(x|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \sigma_i) \tag{15}$$

where, x is the D-dimensional data vector of measurements or features, w_i is the mixture weights and $g(x|\mu_i, \sigma_i \; \forall i = 1, 2, ...M$ are the sub densities (components of the non-Gaussian input), μ_i and σ_i are the mean and covariance respectively. By splitting the incoming PDF in this manner of subcomponents, our information fusion scheme can tackle non-Gaussian distributed data.

Algorithm 3: LEVEL 2: Inference Algorithm

```
Input: Input graph from level 1 G, attacks, system state values
Output: Hypotheses, Control Messages, Network rescheduling
for attacks a_i ∈ A_t = {a_1, ...a_k} & s_k ∈ S do
    while DoS attack do
        compute τ_mati, τ_mad, τ_e2e, τ_sys
        update all
        flag Sensor {s_k} ∈ {S}
    end
    while Wormhole attack do
        compute τ_mati, τ_mad, τ_e2e, τ_sys
        update all
        obtain no. of hops, τ_tn
        flag Sensor {s_k} ∈ {S}
    end
    while Stealthy attack do
        compute State estimate x_t
        infer inconsistent data values
        compute Quantization error ε_q
        update τ_sam
        flag Sensor {s_k} ∈ {S}
    end
    while Replay attack do
        compute State estimate x_t
        infer repetitive data values
        compute residual error ρ
        update τ_sam
        flag Sensor {s_k} ∈ {S}
    end
    determine faulty sensors ← flagged sensors
    recompute τ_sam, τ_mati, τ_mad
    updateDBN
    G ← Ǵ
    Level 2 == TRUE
    return Hypotheses, Control Messages, Network
    rescheduling, information, τ_mati, τ_mad
    END
end
```

V. EXPERIMENTAL SETUP

We made use of both PiccSim [13] and NS2 to simulate the NCS. PiccSim is a simulation software that allows us to model the control plant. This model is the input to the simulator. The sensor network is simulated using NS2. PiccSim is MATLAB based with a feature that can link the control model in MATLAB to the network model in NS2 . The communication between the two is done using specific assigned ports as shown in figure 2.

An inbuilt scheduling method schedules the manner in which the data and control messages are transmitted. The packet transmissions are done over UDP protocol. TCP can be used, but it increases unwanted retransmissions thereby congesting the network. We used JCSTR (Jacketed Continuous Stirred-tank Reactor) as described in [14]. In this simulation, a tank inlet stream is received from another process unit and there is a heat transfer liquid that circulates through the jacket to heat the liquid in the tank. The objective is to control the temperature and the volume in the tank by varying not only the jacket inlet valve flow rate but also the tank outlet valve flow rate.

The control system is modeled first by linearizing the non-linear model (building an appropriate linear model). The sensors are placed at the jacket inlet, tank outlet and to determine the level of the liquid in the tank and also to determine the temperature of the liquid in the tank. In total, 100 sensors are deployed. We loaded the plant model into the simulation GUI by using PiccSim. The plant model is constructed in MATLAB and the m-file is loaded. On another system, which runs NS2, the network topology is loaded with required wireless network parameters like the propagation model, Routing protocol, MAC protocol (MAC CSMA/CA) and the network connection pattern. As, the network connection varies to reroute traffic according to new scheduling policies, different possibilities of network connections are considered in the .tcl file. Our aim was to generate data from a real time simulation, however, due to known issues with network emulation

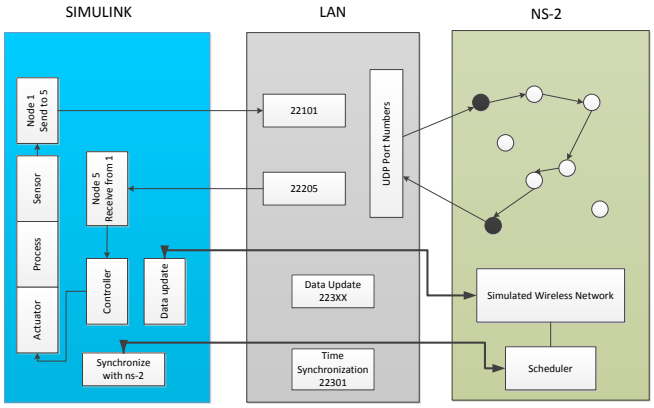

Figure 2. Representation of the PiccSim model

in NS2 and problems with 802.11 in emulations we chose not to. This however does not lessen the correctness of the simulation as experiments were run over two machines, which had to connect to each other over Internet. Synchronization is inbuilt in simulation models in PiccSim. We can however alter this depending upon our requirements. PiccSim uses an inbuilt synchronization protocol. Blocks are pre-built in PiccSim to trigger a subsystem when a packet has/ has not arrived at a node block. The data was analyzed using the an Bayesian network. BayesiaLab [15] was used to build the BN and generate new hypotheses and choose from the existing ones.

VI. RESULTS AND ANALYSIS

Experimental results are obtained after 500 simulation runs performed for each attack. The results are averaged out values for the convenience of analysis.

A. SIMULATION OF LEVEL 1 ATTACKS

Replay attacks: Figure 5 illustrates the average packet reject [1] % in a replay attack. A replay attack is one in which the attacker takes control of a certain section of nodes, captures the data values being sent and replays them. In order to detect such attacks, the control messages as determined by the level 2 of the architecture carry information in them which requests a modification of transmitted value by sensors. This modification in the data is calculated by a function with a correctness only the base station can corroborate. Although an attacker captures a node, he will not have information regarding the function and thus, the attack is detected.

In the 500 simulation runs, for replay attacks, the number of nodes captured, defines the packet reject %. In figure 5, the packet reject % indicates the number of packets that are rejected as a result of a successful attack. The actual attack line depicts the base case of performance when no security mechanism is in place. We compare our algorithm Fusion TVDBN (F-TVDBN) with the Switched Linear Dynamics System (SLDS) [16] and the kernel-weighted TVDBN (k-TVDBN) [17] methods of implementation. F-TVDBN indicates the use of our technique where the rejection ratio is significantly low. This is because, we use Kalman filters to predict the data values at the controller node which feeds the actuators of the control plant with a very close approximation of the actual data. The measurement residual was adjusted based on the output from the inference algorithm about an attack. This approach ensures a gradual degradation in the functioning of the control

[1] packet reject % is the % of packets that are rejected by the IFA or the user as opposed to packet drops which are specific to network behavior

plant rather than working on entirely false data that does not reflect the actual working of the plant as reported by sensors.

Figure 3 shows the accuracy of packet reject % with respect to the number of genuine packets being rejected. The number of genuine packets being rejected is low because of the relative ease of finding out the distortion in the expected pdf of the state estimates. The pdf of the state estimates for a certain section of the WNCS is known a priori. However, it is also known that the pdf of the system state does not repeat itself over several time intervals. The accepted values of the system state is determined by the confidence intervals of the pdf. Injecting random signals into the system will alter the moments of the pdf. Packets that fall within the bounds determined by the moments are considered good packets. However, due to varying noise levels and system faults some good packets may fall outside this range and are considered faulty. In our simulations, for replay attacks, on an average, the total number of packets was 1000 per hour. In which, the average number of faulty packets during various attack strengths were 434 making the average number of good packets to be 566.

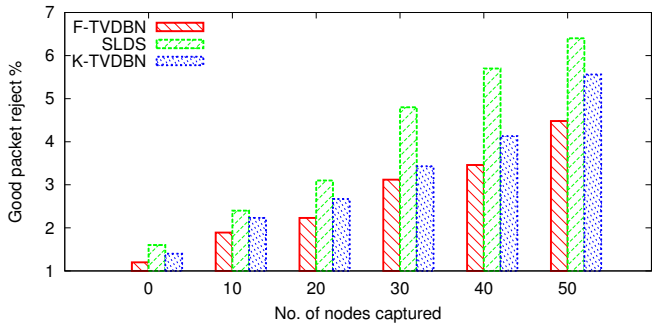

Figure 3. Good packet rejection % in Replay attacks

Stealthy data deception attacks: A similar approach was employed to determine of

the presence of a data deception attack. A stealthy data deception attack works by cleverly deceiving the base station into believing that the values reported are accurate whereas the actual values are modified in such a manner that they appear genuine. An intelligent attacker will try to ensure that the corrupted system state never exceed the measurement residual. This type of attack is used for the objective to steal resources from the plant e.g. power grid theft and water supply theft. We use similar methods of detection employed by [1, 3]. However, in our method, we tighten the error tolerance rate based on the hypotheses generated in level 2 of our information fusion architecture. This error tolerance rate is determined by sensor network aggregation, which makes sure that the error bound condition has to be met by the group of sensors. By doing so, we force an increase in the scope of attack implementation making it difficult for an attacker to attack. This can be seen in figure 6 where, if upto 40% of the nodes are captured, the packet rejection % slowly starts coming down and then increases gradually.

Figure 4 discusses the rejection rate of the number of good data packets in the stealthy deception attack. Based on the discussion in **??**, for varying system noise levels, allowable errors due to bad model design arising from incomplete knowledge of system behavior, the number of genuine packets rejected by our information fusion model for different attack strengths is shown in figure 4. It is assumed that the attacker at any given time does not have complete system knowledge. The attackers partial knowledge is represented in the observation matrix H, the choice of which determines the convergence of the residual value ρ. Adjusting the ρ value to determine the impact of an attack is of principal importance in limiting the scope of an attacker, as then the attacker will have to risk being detected by increasing the severity of the attack. In the figure 4 we can see that with an increase in the number of nodes captured from 20% to 35%, the rejection rate of good packets comes down. This is because of the manner in which the pdf of the residual value is calculated by our fusion algorithm. After 35% of node capture, we simulate the change in the attacker behavior in such a manner that the corrupted residual still converges to the

accepted residual. As we can see, our algorithm even under such extremely hard detection conditions performs much better than the other learning methods. This is attributed to the fact that our algorithm takes the overall impact of the parameters and features rather than just the pdf of the residual and the known system model. In our simulations, for the stealthy data deception attack, n an average, the total number of packets was 1000 per hour. In which, the average number of faulty packets during various attack strengths were 443 making the average number of good packets to be 557.

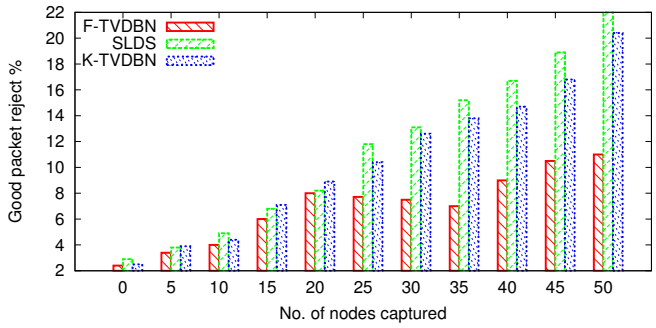

Figure 4. Good packet rejection % in Data Deception attacks

Before we go on to analyze the other attacks, we would first like to describe how for different initial conditions different τ values were chosen. Figure 7 is an illustration of how under changing τ values, the control packet size is altered and how the packet drop % varies. From theorem 1, for different values of λ of the protocols such as Round Robin (RR) and Try-Once Discard (TOD), the values of initial conditions ϕ_ℓ, where, $\ell = 0, 1$ are chosen based on the number of nodes used in the simulation. For instance $\lambda_{TOD} = \sqrt{(n-1)/n}$, where, n is the number of nodes. Based on this, for different initial conditions of the NCS, we obtained different τ values. For each of those τ values, we altered the packet size of the control messages as can be seen from figure 7. We use this information in the analysis of other attacks.

Denial of Service attacks: In a DoS attack, the service is thwarted by either infinitely rerouting packets in the network or by disrupting the flow of traffic or by causing the network timeouts etc. In our implementation, as we do not use TCP, network timeouts do not occur. We used our F-TVDBN to output an appropriate network rescheduling policy which reroutes the packets from malicious nodes. As we can see from the results of avg. packet drop rate in figure 8, with an increase in the number of nodes, in F-TVDBN, the number of packet drops is less as compared to the one which uses Hidden Markov Model (HMM) as an inference algorithm. Both these are compared to the base case where no rescheduling takes place. We observed that even after employing our scheme, in many cases the packet drop rate is still relatively low even after the number of nodes captured is >50. The end-to-end delay helps us determine the sampling rates and the sampling time intervals for the next time instant.

In order to tackle the effect of DoS attack (jamming attack in this paper), our fusion architecture recomputed the values of τ_{mati} and τ_{mad} to maintain operational stability. Figure 9 shows the delay in seconds for the number of packets transmitted over different time slices. We introduced the attack in the system during our simulation and two different time instants. If no delay compensation technique is employed, the controller must receive an ith sample from the plant before computation of $i+1^{th}$ control signal to ensure stable operation. Hence, total delay must satisfy the condition that $\tau_{delay} < \tau_{mad}$. This decision allows us to make appropriate conclusions about the operational stability of the plant and subsequently modify the hypotheses to better suit the WNCS requirements. Sampling rate can also be reduced at the cost of a slight loss in the data fidelity. This can be done to save energy on the nodes under no attack. Once the attack is detected and mitigated, the sampling rates are adjusted to suit the needs.

Wormhole attacks: As we can see from the figure 11, the % of packet drops is almost same as for a denial of service attack. The number of nodes captured governs the packet loss. The end-to-end delay in a wormhole attack is more pronounced as the nodes forward

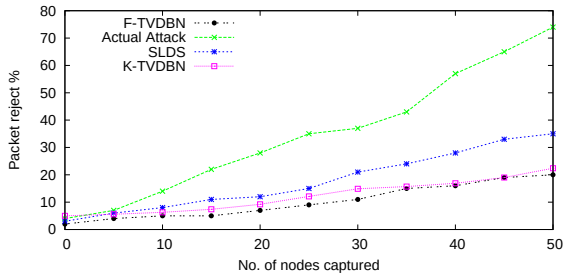

Figure 5. Packet rejection vs Nodes captured in Replay attacks

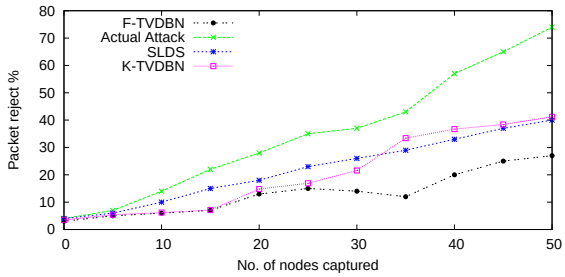

Figure 6. Packet rejection vs Nodes captured in Data Deception attacks

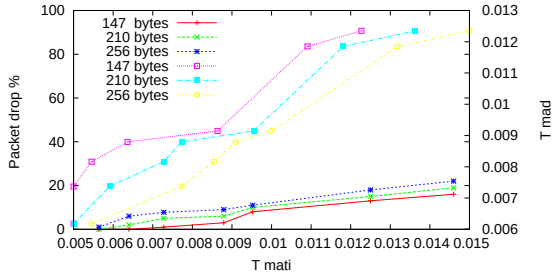

Figure 7. Packet drop % vs ctrl. pkt size under different τ_{mati} and τ_{mad}

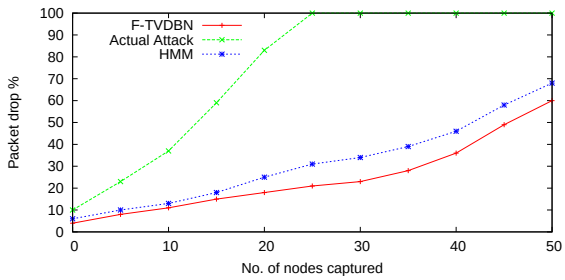

Figure 8. Packet drop vs Nodes captured in DoS attacks

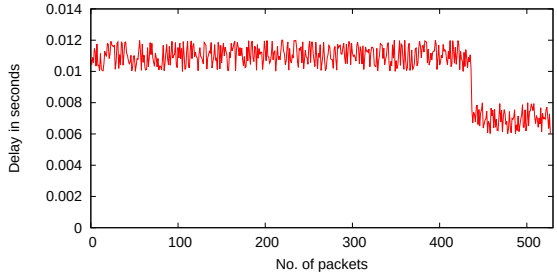

Figure 9. End-to-End delay in DoS attack during first time slice

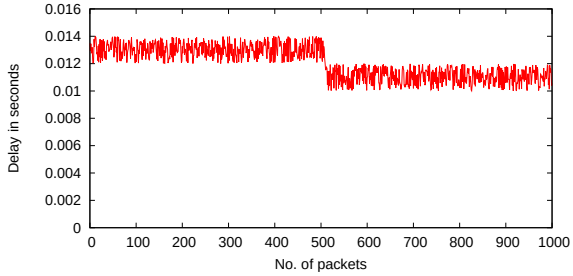

Figure 10. End-to-End delay in Dos attacks during second time slice

the packets to destinations that are not meant to be, so an increase in the number of nodes captured will also affect the end-to-end delay. The packet drop % was compared against the use of a HMM. As we can see from figures 12 and 13, during the second attack, our scheme attempted to reduce the network delay significantly by upholding the condition $\tau_{delay} < \tau_{mati} < \tau_{mad}$ thereby ensuring operational stability of the plant. However, the values of τ_{mati} and τ_{mad} were updated to suit the longer delays in the network. This operation acts as the gradual degradation of the network rather than the network suddenly disrupting

thereby causing the system to fail.

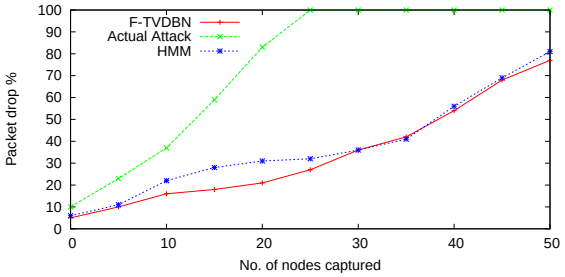

Figure 11. Packet drop vs Number of Nodes captured in Wormhole attacks

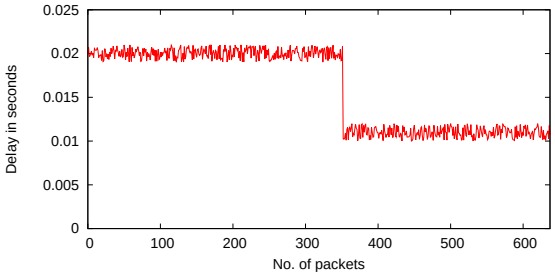

Figure 12. End-to-End delay in Wormhole attacks during first time slice

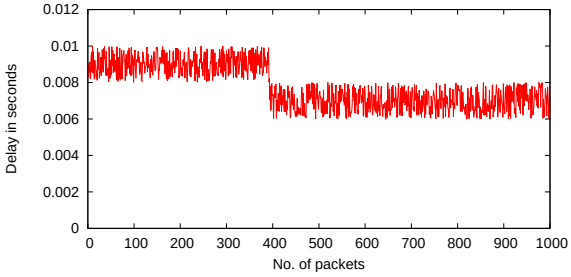

Figure 13. End-to-End delay in Wormhole attacks during second time slice

Collaborative attacks: Figure 14 is a graphical representation of how the number of nodes captured affects the packet drop % in our scheme vs HMM vs in actual attack. We can see that by the time 50% of the nodes are captured, in collaborative attacks, the number of packet drops goes as high as 88% in our F-TVDBN scheme and 94% in the HMM scheme.

In the collaborative Replay and Data deception attacks, there was barely any difference in terms of graphical representation as both of them fall under stealthy attacks and perform similarly in the study of packet reject%.

B. LEVEL 2 OF INFORMATION FUSION: BAYESIAN NETWORK FOR HYPOTHESES GENERATION

With the information obtained from the simulations, we computed the conditional probabilities to determine the presence of a standalone, or, a collaborative attack. These probabilities are continuously computed during run time and the updates are sent as input to the TVDBN. A set of hypotheses was initially generated according to a priori experiments that were run offline. This was done in order to enable us to choose from a preset hypothesis and take necessary measures upon the detection of an event. As and when the TVDBN was updated, we had a more defined and clear idea about the system behavior. This information was incorporated into the determination of new hypotheses. Sometimes, it is possible for the attacker to perform collaborative attacks. By doing so, the attacker can confuse the system administrator and get away with without being detected. We considered such cases also as listed in the joint probability values in the Table 2. To build such causal relations among values, we used BayesiaLab (Bayes network tool). Table 1 provides us with the concerned probabilities with respect to different parameters. This table indicates the effect of (probabilistically) the various network parameters on the system. The probability of an attack was computed according to this. According to this information, and by including the effect of unknown parameters, we derived the conditional probability values.

Attack 1 (A1) - Replay Attack

Attack 2 (A2) - Deception Attack

Attack 3 (A3) - DoS Attack

Attack 4 (A4) - Wormhole Attack

Parameter (P1) - Packet Drops

Table 1. Parameters and Probabilities of Attacks

Parameters	Wormhole attack	DoS attack	Deception attack	Replay attack
Packet Drops	0.89	0.93	0.42	0.37
System Delay	0.94	0.96	0.33	0.31
Communication Delay	0.95	0.96	0.29	0.21
Inter-arrival time delay	0.92	0.92	0.18	0.22
Quantization Error	0.46	0.32	0.93	0.92
Inconsistent data	0.41	0.56	0.91	0.82
Repeated data	0.13	0.18	0.89	0.94

Parameter (P2) - System Delay

Parameter (P3) - Communication Delay

Parameter (P4) - Inter-arrival time delay

Parameter (P5) - Quantization Error

Parameter (P6) - Inconsistent data

Parameter (P7) - Repeated data

Other factors and Unknowns (U) - Probability = 0.1 (assumed that there is a 10% chance of deviation from normal behavior). Now, from the experimental observations, we calculated the conditional probability values as shown in Table 2, where $*$ indicates collaborative attacks.

Hence, according to these probability values, the Bayesian network is generated. We define preset hypotheses viz.

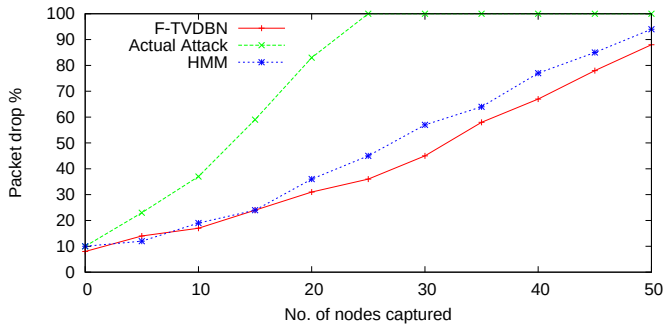

Figure 14. Packet drop vs Nodes captured in collaborative attacks

Table 2. Parameters and Conditional Probabilities

Attack/parameters	P1	P2	P3	P4	P5	P6	P7	Total Probability
A1	0.37	0.31	0.21	0.22	0.92	0.82	0.94	0.94
A2	0.42	0.33	0.29	0.18	0.93	0.91	0.89	0.93
A3	0.93	0.96	0.96	0.95	0.32	0.56	0.18	0.96
A4	0.89	0.94	0.95	0.92	0.46	0.41	0.13	0.95
$A1 \cap A2 *$	0.37	0.32	0.24	0.21	0.96	0.93	0.91	0.94
$A2 \cap A3 *$	0.94	0.95	0.89	0.91	0.58	0.49	0.17	0.96
$A3 \mid A2 \cap A4 *$	0.87	0.88	0.92	0.89	0.30	0.33	0.29	0.89
$A4 \mid A2 \cap A3 *$	0.85	0.81	0.87	0.84	0.10	0.19	0.12	0.88

H_0 = Attack 1 is accepted as it is; increase the number of function based value requests and reroute traffic

H_1 = Attack 2 is accepted as it is; tighten the error tolerance

H_2 = Attack 3 is accepted as it is; change the network scheduling and routing and increase the sampling rate

H_3 = Attack 4 is accepted as it is; change the network scheduling and routing and increase the sampling rate

H_4 = Attack 1 & Attack 2 could take place; repeat measures from hypotheses 1 and 2

H_5 = Attack 3 & Attack 4 could take place; repeat measures from hypotheses 3 and 4

H_6 = No attack took place; the disruption could be a false alarm, reschedule the network

and run Kalman filtering to reset the network

According to the experimental results from 500 simulation runs, hypothesis H_0 was selected 88.4% of the time when there was a replay attack actually took place. H_1 was selected 88.7% of the time when a deception attack took place. H_2 was selected 96.5% of the time when a DoS attack took place and H_3 was selected 92.13% of the time when the network was subjected to wormhole attacks. H_4 was selected 94.3% of the time when either attack took place which shows the efficiency of our approach. H_5 was selected 97.2% of the time when either DoS or wormhole attacks took place. However, in 9.8% of the simulations, we were unable to differentiate between the two attacks. Inspite of this setback, we were able to uphold the system stability. The inaccurate determination (9.8%) was observed owing to both the system noise and lack of information. False positives were recorded in 2.2% of the total simulation runs i.e., in 11 out of 500 runs when our system reported an attack when no attack existed. False negatives were recorded during 5.4% of the time i.e. in 27 of the total 500 runs. Of the 27 runs, in 14 runs when we simulated the deception attacks slightly over the accepted bound, our scheme could not detect any attack. In the other 13 runs, upon altering the system noise (randomizing the noise input) we could not detect the presence of an adversary. This throws light upon the effect noise on the system stability and security.

VII. RELATED WORK

Vast literature exists on both the performance and stability of NCS. Although, most of this work is concentrated on scheduling and security algorithms for wired systems, research about the use of wireless sensors has grown. The NCS systems which use the wireless sensors to make them WNCS have many advantages, but with limitations. The information sent out by them can be easily spoofed, read, rerouted indefinitely etc. This causes severe damage to the control plant. Various studies [1, 2, 3, 1, 5] have dealt with the encryption schemes in WNCS. One scheme [2] makes use of DES to encrypt the data messages and thus, provides confidentiality. The study claims that use of such a technique gives enough time before the attacker can actually read the message before the message becomes obsolete. Although this claim holds true, encryption techniques alone, however, cannot make a control plant secure. Studies such as [17, 18] discuss both scheduling issues and stability analysis and provide effective ways to tackle their scheduling. In [13] effective data rates were sustained to maintain the stability of the system under different closed loop and open loop conditions. This process is completed by feeding the control system with both preset functional and control states that govern the stability of the system as required does this process. The authors aim to reduce the need of excessive data rates as required by the control plant.

So far, studies pertaining to security in WNCS focused on stand alone attacks. Solutions for the detection and prevention of collaborative attacks have not been proposed. For instance, an attacker can employ both, an attack on the wireless sensors and the control plant e.g. both DoS attack and Replay attack can take place simultaneously. This might render the sensor/control messages to be in transit for a period long enough that a human observer can overlook it as just an improper functioning or as just a schedule miss. This can prove to be very costly for the smooth operation of a control system. In [1] deception

and DoS attacks against an NCS are introduced, and, for the latter attack, a countermeasure according to semi-definite programming is proposed. In [2, 3], false data injection attacks against static state estimators are introduced ad studied. It is shown that undetectable false data injection attacks can be designed even when the attacker has limited resources. In a similar fashion, stealthy deception attacks against the SCADA systems are studied, among others, in [4, 5]. In [6], the effect of covert attacks against networked control systems is investigated. Specifically, a parameterized decoupling structure allows a covert agent to alter the behavior of the physical plant while remaining undetected from the original controller. In [7], a resilient control problem is studied, in which control packets transmitted over a network are corrupted by a human adversary. This kind of attack is an insider attack. A receding-horizon Stackelberg control law is proposed to stabilize the control system despite the attack. Recently, the problem of estimating the state of a linear system with corrupted measurements has been studied [9]. More precisely, the maximum number of faulty sensors that can be tolerated is characterized, and a decoding algorithm is proposed to detect corrupted measurements. Finally, security issues of some specific cyber-physical systems have received considerable attention, such as power networks, linear networks with misbehaving components and water networks [3, 7].

Most recently authors in [26] proposed a distributed security framework for heterogeneous WSNs which is used largely for monitoring. This approach closely resembles our proposed scheme in that their framework is also a security framework which takes into consideration multiple attacks, but does not provide models used to detect individual or collaborative attacks. Additionally, our algorithm goes a step ahead and generates decisions which are crucial in real-time applications like NCS. Our approach differs from them in the sense that we do not just monitor, but also provide the backbone to build an automated process to handle real-time applications.

In summary, our proposed approach is different from all the mentioned approaches as none of the solutions proposed are comprehensive except [26], i.e. they can only be used

in the detection of stand alone attacks. In the wake of growing need for an fully secure and stable NCS, these solutions fall short. Our proposed fusion architecture performs admirably by giving a unified solution to both stand alone and collaborative attacks. Furthermore, we achieve this without using any encryption technique thereby reducing the overhead of communication.

VIII. CONCLUSIONS

Through this paper we have introduced an information acquisition and fusion scheme for Wireless Networked Control System (WNCS), where, both feature extraction and decision making are used in securing and stabilizing the system. We provided theoretical bounds of time delays in order to ascertain system stability. In order to validate the efficacy of our approach, we conducted several experiments using PiccSim and NS2. Through the analysis of the results we obtained we understood that stabilizing a WNCS under security attacks cannot be completely guaranteed. However, by using our approach we have shown that we can mitigate the effects of security attacks and still provide the WNCS with operational stability. Our approach improves on existing inference techniques as discussed in the analysis section. We showed using a Dynamic Bayesian Network, how changes in the network behavior can be efficiently observed and decisions can be made. Based on these decisions, the control messages had been altered to suit the requirements of WNCS.

REFERENCES

[1] A. Teixeira, S. Amin, H. Sandberg, K. Johansson, and S. Sastry, "Cyber security analysis of state estimators in electric power systems," in *Decision and Control (CDC), 49th IEEE Conference on*, 2010, pp. 5991–5998.

[2] Z.-H. Pang, G. Zheng, G.-P. Liu, and C.-X. Luo;, "Secure transmission mechanism for networked control systems under deception attacks." IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), Mar. 2011, pp. 27–32.

[3] A. Teixeira, H. Sandberg, and K. H. . Johansson, "Networked control systems under cyber attacks with applications to power networks." American Control Conference (ACC), 2010, pp. 3690–3696.

[4] S. Amin, A. Cardenas, and S. Sastry, "Safe and secure networked control systems under denial-of-service attacks," *Hybrid Systems Computation and Control*, vol. 5469, pp. 31–45, Apr. 2009.

[5] R. A. Gupta and M.-Y. Chow, "Performance assessment and compensation for secure networked control systems." 34th Annual IEEE Conference of Industrial Electronics, IECON, Nov. 2008, pp. 2929–2934.

[6] W. P. H. Heemels, A. Teel, N. Van de Wouw, and D. NesÌŇicÌĄ, "Networked control systems with communication constraints: Tradeoffs between transmission intervals, delays and performance," vol. 55, no. 8, 2010, pp. 1781–1796.

[7] D. Liberzon and D. Nesic, "Input-to-state stabilization of linear systems with quantized state measurements," *Automatic Control, IEEE Transactions on*, vol. 52, no. 5, pp. 767–781, May 2007.

[8] U. M. L. Repository, `http://archive.ics.uci.edu/ml/datasets/Water+Treatment+Plant`.

[9] A. Tucker and X. Liu, "A bayesian network approach to explaining time series with changing structure," *Intelligent Data Analysis*, vol. 8, no. 5, pp. 469–480, Oct. 2004.

[10] S. H. Nielsen and T. D. Nielsen, "Adapting bayes network structures to non-stationary domains," *International Journal of Approx. Reasoning*, vol. 49, no. 2, pp. 379–397, Oct. 2008.

[11] L. Song, M. Kolar, and E. P. Xing, "Time-varying dynamic bayesian networks." in *NIPS*, 2009, pp. 1732–1740.

[12] X. Xuan and K. Murphy, "Modeling changing dependency structure in multivariate time series," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 1055–1062.

[13] J. H. Taylor, H. Ibrahim, and J. Slipp, "A safe communication scheme for an intelligent wireless networked control system coordination agent." IEEE International Conference on Systems Man and Cybernetics (SMC), Oct. 2010, pp. 3068–3073.

[14] J. H. Taylor and J. Slipp, "An integrated testbed for advanced wireless networked control systems technology," in *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2010, pp. 2101–2106.

[15] BayesiaLab, `http://www.bayesia.com/en/products/bayesialab/tutorial.php`.

[16] E. Fox, E. B. Sudderth, M. I. Jordan, and A. Willsky, "Bayesian nonparametric inference of switching dynamic linear models," *Signal Processing, IEEE Transactions on*, vol. 59, no. 4, pp. 1569–1585, 2011.

[17] G. C. Walsh, H. Ye, and I. L. G. . Bushnel, "Stability analysis of networked control systems," *IEEE Transactions on Control Systems Technology,*, vol. 10, no. 3, pp. 438–446, May 2002.

[18] S. Hariharan and N. B. Shroff, "Deadline constrained scheduling for data aggregation in unreliable sensor networks." International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), May 2011, pp. 140–147.

[19] Y. Liu, M. K. Reiter, and P. Ning, "False data injection attacks against state estimation in electric power grids." ACM Conference on Computer and Communications Security, Nov. 2009, pp. 21–32.

[20] Y. Mo and B. Sinopoli, "Secure control against replay attacks," in *Communication, Control, and Computing, Allerton. 47th Annual Allerton Conference on*, 2009, pp. 911–918.

[21] A. Teixeira, S. Amin, H. Sandberg, K. Johansson, and S. Sastry, "Cyber security analysis of state estimators in electric power systems," in *Decision and Control (CDC), 49th IEEE Conference on*, 2010, pp. 5991–5998.

[22] S. Amin, X. Litrico, S. S. Sastry, and A. M. Bayen, "Stealthy deception attacks on water scada systems," in *Proceedings of the 13th ACM international conference on Hybrid systems: computation and control*, ser. HSCC, 2010, pp. 161–170.

[23] R. Smith, *A decoupled feedback structure for covertly appropriating network control systems*. in IFAC World Congress, Milan, Italy, Aug. 2011.

[24] M. Zhu and S. Martinez, "Stackelberg-game analysis of correlated attacks in cyber-physical systems," in *American Control Conference (ACC)*, 2011, pp. 4063–4068.

[25] H. Fawzi, P. Tabuada, and S. Diggavi, "Secure state-estimation for dynamical systems under active adversaries," in *Communication, Control, and Computing (Allerton), 49th Annual Allerton Conference on*, 2011, pp. 337–344.

[26] M. Valero, S. S. Jung, A. Uluagac, Y. Li, and R. Beyah, "Di-sec: A distributed security framework for heterogeneous wireless sensor networks," in *INFOCOM, Proceedings IEEE*, march 2012, pp. 585 –593.

III. INFORMATION FUSION ARCHITECTURE FOR VARIABLE-LOAD SCHEDULING IN A CLOUD-ASSISTED CYBER PHYSICAL SYSTEM

Brijesh Kashyap Chejerla, Sanjay K Madria

Department of Computer Science,

Missouri University of Science and Technology, Rolla, Missouri 65401

This paper addresses the problem of devising an effective task scheduling algorithm which facilitates data processing of a Cyber Physical System (CPS) under bounded latency under bursty or lossy traffic. Task scheduling traditionally caters to real-time systems where a feedback loop does not exist allowing the serviced application to be independent of the inputs from the server. However, owing to the nature of a near real-time CPS, such liberties cannot be entertained. Additionally, the advent of big data in CPS has necessitated the use of Cloud Computing as a scalable and cost effective alternative. Task scheduling in such CPSs, where inputs from the Cloud complete the feedback loop is a major research issue. Therefore, in this paper, we propose a multi-layered information fusion architecture which provides for such a task scheduling mechanism by also accommodating both traffic bursts and packet losses. Our scheduling algorithm ensures that the overall latency always remains under an acceptable upper bound as required by the CPS application. We examine the performance of our algorithm on an Amazon cloud using a real data set.

I. INTRODUCTION

Cyber-physical systems (CPS) exhibit sophisticated system behavior that results from coupling the physical components in classical engineering systems with the cyber components that process information. The sophistication lies in the time-varying manner of the role and existence of the number of sub-components that constitute a CPS. The research challenge with such dynamic and complex systems is to tightly couple the physical and cyber components through available physical attributes and acquired information. The solution to this challenge is a decision support system which tracks, predicts and addresses the needs of the system by outputting control decisions which closes the feedback loop of a CPS. Such a decision support system cannot be made available through the on-board processors on the physical components and requires a more scalable infrastructure such as Cloud Computing [1].

Some applications such as *transportation systems and smart grids* are considered as complex hybrid CPSs. These systems cannot always be converged into a finite and defined set of initial configurations owing to constraints such as lack of absolute knowledge about system behavior. To solve this problem, a feature extraction and selection mechanism that draws implicit information from the limited knowledge is essential. It helps in understanding the system behavior better and laying a strong basis for future decision making. Once the features are extracted and the constraints delimited, the next challenge is to efficiently process the information for quick dissemination over the entire network. This requires an efficient prediction mechanism that determines the amount of processing power and buffer space required for the tasks corresponding to incoming traffic stream.

CPS applications comprise a number of devices such as sensors and actuators distributed over a large geographical area. Increasingly, sensors in CPS applications are generating large volumes of data with varying processing requirements and highly variable

traffic. Traditionally, the traffic model of a CPS is characterized by an *ON/OFF* behavior [2]. An *ON/OFF* traffic model in itself is not self-similar and does not support bursty traffic. However, an aggregated traffic trace that is a superposition of a large number of independent *ON/OFF* traffic from different sources with heavy tail exhibits a second order *Self-similar* behavior. The degree of self-similarity in the traffic is determined by the Hurst parameter H and if $0.5 < H < 1$ the traffic exhibits *Long Range Dependence (LRD)*. Self-similar traffic contains periods of bursty traffic which is hard to predict. Serving such a traffic with low latency is another research challenge.

Scheduling of data generated by CPS has utmost priority. Conventional CPSs cannot handle the growing requirements of big data because of constraints such as limited on board processing, network and bandwidth constraints which are major bottlenecks for tasks such as: data grouping, aggregation and inter-dependent processing. This problem becomes more severe when the traffic and data are characterized by burstiness. Obtaining bounded latency under such constraints is challenging. Several schemes exist [3, 4, 5, 6, 7, 8] for applications that distribute the incoming traffic onto the virtual machines on the cloud without being required to close the feedback loop. However, they fall short when it comes to minimizing latency that arises in time-critical CPS applications with a feedback loop. Also, task scheduling is a decision problem which is NP-Hard and in some cases NP-Complete and thus, has to be solved by approximating it to a known and polynomially solvable algorithm.

To address these research issues, we propose an Information Fusion Architecture (IFA). We first devise an effective feature selection algorithm that chooses the features that most appropriately describe the initial configuration of the CPS. This approach is iterative, where, features are added and removed from a pool of features in $O(nlogn)$ time. Next, our IFA solves the above mentioned problems of obtaining bounded latency under burstiness by using a causal graphical model. Inputs from the feature extraction and selection algorithm are used to construct a causal graph, the output of which are multiple snapshots that describe system behavior under varying network parameters. Finally, we propose a

scheduling algorithm which uses the causal graph to schedule the tasks onto the virtual machines on the cloud. We test the scalability of our approach by varying the data size of a real CPS application. The results obtained showed us that under varying traffic burstiness, different queue sizes and memory allocations on the VMs, our algorithm is effective in keeping the latency requirements under bounds.

II. PROBLEM STATEMENT

Formally, a set of sensors $S = \{s_1, s_2, ...s_n\}$ on a CPS record observations $O = \{o_1, o_2, ...o_n\}$ over a period of time $\{t_1, t_n\}$. These observations need to be processed in a timely manner and send back the information about the subsequent events from the decisions taken by an automated architecture. Upon obtaining sensor observations as grouped data, the problem becomes two fold - (i) feature selection to determine the most useful features that aid in decision making, and, (ii) Task prioritization based on QoS requirements of the CPS. Next, to process these tasks, an effective scheduling algorithm has to be devised that keeps the latency under bounds by determining the required number of virtual machines $VM = \{vm_1, vm_2, ...vm_n\}$. The proposed algorithm should handle the complications of dependent tasks which categorized it to be a NP Complete decision making problem.

III. RELATED WORK

Most traffic models for vehicular and transportation systems, communication and biological networks based on queuing theory [9, 10, 11, 12] rely on the assumption of having a Poisson packet arrival process and a random service time that is independent of the arrival process. Queuing models can be used to manage the load on servers based on the incoming traffic streams by computing the average processing times, average waiting times in queues etc. based on the traffic information like inter-arrival times. However, not all queuing models are suitable to deal with the bursty arrival traffic. In [9, 10, 11, 12], the traffic is self-similar and authors claim that the Hurst exponent is enough to capture the characteristic of the traffic. As opposed to this view, authors in [2] claim that the fractal nature of a traffic trace from heterogeneous source cannot be captured by just the Hurst exponent. In [2], it is shown that the moments associated with various traffic processes e.g. the packet arrival, average waiting times etc. follow a scaling behavior. In [13], query processing is done using queuing theory to process the queries stacked in the buffers in appropriate time. Buffer width is chosen based an optimization scheme to design the query algorithm. The buffer optimization scheme does not consider bursty packet arrivals. The solution provided in [13] deals with a different problem altogether as traffic that the buffers are designed for, do not exhibit any self-similar behavior with long range dependence.

In order to better model the traffic exhibiting a self-similar behavior, many prediction schemes have been proposed[14, 15, 16]. These schemes have their own short comings as shown in terms of the accuracy and the latency with which they predict the incoming traffic. Traffic modeling in CPS [17] and its related applications has not been studied extensively. However, few studies that deal with smart grids have observed the traffic to be long range dependent and have proposed solutions that use virtual queues with min-max queuing [18] and the use of Mellin transforms [19] to show the effect of long-range

dependence on the loss rate of energy tasks in a queued energy model. However, these models do not provide an integrated solution that solves the NP-hard decision problem of scheduling the most appropriate task into a queue for further processing. These papers use on-site processing centers and thus, the latency requirements are met. Works that tackle the problem of query processing in transportation systems, earthquake monitoring systems etc. so far have not used the capabilities of cloud computing to solve the problem of low elasticity and limited processing. In [20], the authors talk about need for the use of a Cloud computing architecture in CPS applications. However, the use of Cloud computing poses problems that pertain to latency in data dissemination.

In order to effectively solve this problem, an efficient load balancing mechanism built around task scheduling is essential. In [3, 4, 5, 6, 7, 8], load balancing mechanisms have been proposed based on scheduling. However, scheduling in these schemes tackles the scheduling of the virtual machines to do the job/ task processing; not the scheduling of incoming tasks. In works such as [21] authors propose an algorithm that tries to solve the decision problem of task scheduling based on priority using ant colony optimization. All the above mentioned algorithms provide solutions to individual problems in processing information from an application. Our approach differs from them in that we provide a unified solution that uses the concepts of prediction in self-similar traffic streams, a causal graph structure for making decisions to solve the NP-complete problem of task scheduling and load balancing in cloud computing. Using our fusion architecture, we provide a CPS application with elastic processing capabilities with minimum latency.

IV. PRELIMINARIES

A. SELF-SIMILARITY

The notion of self-similarity [22] in traffic traces can be simply explained as *the phenomenon where a property of a sample of a traffic trace is preserved with respect to scaling in both space and/or time, i.e., if a traffic is self-similar, a part of it when magnified will look similar to the shape of the whole*.

Mathematically it is defined as follows:

Let $X = (X_t : t = 0,1,2,...)$ be a covariance stationary (wide-sense stationary) stochastic process; i.e., a process with constant mean $\mu = E[X_t]$, finite variance $\sigma^2 = E[(X_t - \mu)^2]$, and an autocorrelation function

$$r(k) = E[(X_t - \mu)(X_{t+k} - \mu)]/E[(X_t - \mu)^2] \tag{1}$$

where $(k = 0,1,2,...)$. In particular, X has an autocorrelation function of the form

$$r(k) \sim c_1 k^{-\beta}, as\ k \to \infty \tag{2}$$

where, $0 < \beta < 1$ and $c1$ denotes a finite positive constant. For each $m = 1,2,3,...$, let $X(m) = (X_k^{(m)} : k = 1,2,3,...)$ denote a new time series obtained by averaging the original series X over non-overlapping blocks of size m. That is, for each $m = 1,2,3,...$, $X(m)$ is given as:

$$X_k^{(m)} = 1/m(X_{km-m+1} + ... + X_{km}), (k \geq 1) \tag{3}$$

For each m, if the aggregated time series $X^{(m)}$ defines a covariance stationary process,

$r^{(m)}$ denotes the corresponding autocorrelation function. The process X is called *exactly second-order self-similar* with self-similarity parameter $H = 1 - \beta/2$ if the corresponding aggregated processes $X^{(m)}$ have the same correlation structure as X, i.e., the autocorrelation for the aggregated process $r^{(m)}(k) = r(k)$, $\forall$m= $1, 2, ...$ and $k = 1, 2, 3, ...$ In other words, X is exactly self-similar if the aggregated processes $X^{(m)}$ are indistinguishable from X (in general, with respect to the second order statistical properties), e.g. fractional Gaussian noise (fGn) with parameter $1/2 < H < 1$ [22]. A covariance stationary process X is called asymptotically (second-order) self-similar with self-similarity parameter $H = 1 - \beta/2$ if $r^{(m)}(k)$ if it is in agreement asymptotically (as $m, k \to \infty$) with the correlation structure $r(k)$ of X as shown in equation 2. One can obtain the degree of self-similarity by employing the methods as specified in [22].

B. QUEUING THEORY FOR SELF-SIMILAR TRAFFIC

Queuing theory has been extensively used in network analysis and consequently, in modeling the self-similar and LRD nature of a traffic. Investigating queuing behavior with LRD input traffic is important as it provides the theoretical boundaries which limit the amount of traffic that can be served on a server. Information obtained from a queuing model is useful in determining the load on the server at a given point of time. Among the several queuing models used for network performance analysis [9, 10, 11, 12], the one that services self-similar traffic appropriately is the $MAP/G/M/k$ model. This is used in modeling the burstiness in incoming traffic traces; best suited for modeling heavy tailed distributions like the Pareto distribution.

For any queuing system, the utilization ρ is defined as $\rho = \frac{\lambda}{\mu * c}$, where, λ is the arrival rate, μ is the mean and c is the number of servers ($c \geq 1$). Utilization is always required to be ≤ 1 for system stability. The average time spent in the queue (W_q) for a $MAP/G/M/k$

queuing system is

$$W_q = \frac{\delta}{\mu(1-\delta)} \tag{4}$$

where, δ is the unique root of $\delta = A^*(\mu - \mu\delta)$ in the range of $0 < \delta < 1$ and $A^*(s)$ is the Laplace transform of the probability density function for the inter-arrival time. This information is used in the feature extraction and decision making process to determine the waiting times in the queue for an incoming traffic trace. The waiting times are in turn used in determining the latency of the system. As we mentioned earlier, the bounds on the QoS latency are determined to keep the CPS stable. These bounds are application dependent, thus depend on the incoming traffic stream, and the total service time on the virtual machines on the cloud. Upon using any scheduling algorithm, there is a delay of τ_d. This delay encompasses $\tau_{communication}$ and $\tau_{processing}$. In other words, the CPS gets back the control signals after τ_d time units. However, there is a bound on the total delay that can be allowed for the system to be stable. Delay $\tau_d \leq \tau_{mad}$ where, τ_{mad} is the Maximally Allowable Delay (MAD). To put both τ_{mati} and τ_{mad} into perspective, we can state the as follows:

$$0 \leq \tau_d \leq \min\{\tau_{mad}\} \tag{5}$$

Where, τ_{mad} is computed as $\tau_{mad} = W_q + \frac{1}{\lambda * \mu} + \tau_d$. Here, τ_d is the network delay in communication or possible packet drops over the network. Our aim is to recursively compute τ_{mad} for the variable traffic streams depending on the long range dependence of the traffic which governs the tail length and the queue size (queue length). The queue length determines the number of incoming messages that can be processed, the number of packets that are dropped owing to poor queuing and the number of servers to be allocated to serve the information/ data sent over the network. In order to theoretically determine the upper

and the lower bounds for the tail distributions, [23] determine them using a generalized Schilder's theorem, as are given by:

$$P(Q \geq x) = exp(\frac{-1}{2}\frac{-x+(n-m){t_x}^2}{am{t_x}^2H+bn{t_x}^2H}) \tag{6}$$

$$P(Q \geq x) \geq \overline{\Phi}(\frac{-x+(n-m)t_x}{\sqrt{am{t_x}^2H+bn{t_x}^2H}}) \tag{7}$$

where, m is the mean traffic arrival rate $E(\lambda)$, a is the variance coefficient which determines the long range dependence and self-similarity, H is the Hurst parameter, n is the required service capacity to serve the load and x is a random variable determining the queue length. Based on these bounds, we calculate the queue length (queue size in terms of memory occupied) and the latency achieved in serving a queue of such a size.

V. INFORMATION FUSION ARCHITECTURE

A. OVERVIEW

We propose a multi-layered information fusion architecture as shown in Figure 1. The decisions outputted from the architecture become the feedback loop which completes the proposed cloud assisted CPS.

- Level 1: Lays emphasis on the information acquisition through Feature Selection (FS).

Here, we run our feature selection and update process algorithm which corresponds to FS as shown in Figure 1. Using FS we choose the most appropriate features such as the network characteristics of the incoming traffic trace to establish a preliminary setting that models the traffic. Features such as self-similarity and long Range Dependence (LRD) in the traffic trace are also determined to complete the service requirements of the traffic. Additionally, using a probabilistic traffic arrival determination method, we predict the burstiness of the incoming traffic in order to determine the number of Virtual Machines (servers) that could be needed. Such thorough profiling of the incoming traffic is done to determine the choice of a queuing model.

In a dynamically changing CPS comprising of heterogeneous sub-systems, traffic patterns and service requirements will not remain constant. Predicting such changes and modeling such a traffic is a major challenge. For e.g, LRD cannot be serviced in the same manner as a renewal process as LRD is not independent and identically distributed (IID). Furthermore, renewal process cannot handle traffic burstiness which exists when the traffic trace is self similar. Also, packet arrival times may not be represented accurately due to packet drops and/or network delays. In order to tackle these issues efficiently, an appropriate choice of the queuing model is of utmost importance. In our architecture, the values of

the Kendall's notation of the queuing model change with the feedback from the decision maker D as shown in Figure 1. The choice of the queuing model determines the service times and the overall latency which are used as inputs in choosing appropriate snapshots $(s_i, s_j, s_k, s_l, s_m)$ for *Level 3* processing. The features of the traffic that are selected from the Feature Selection algorithm are used to construct the causal graph in *Level 2*.

- Level 2: Consists of a causal graphical model to evaluate causal relationships among several network parameters.

These causal relations are a result of correlations among various network parameters selected from *Level 1* such as Inter-Arrival Times (IAT), elements of LRD such as: self-similarity (Hurst parameter H) and shape parameter α of the traffic distribution, burstiness of the next incoming traffic stream over a time window (B.int) and traffic arrival rate λ (with packet drops and delays included in this node of the graph). The causal relations are also among parameters on the processing units such as queue sizes (Q Size), buffer occupancy on the VMs (Buff. space), utilization etc. as depicted in level two of Figure 1. These causal relations are represented as a Bayesian Network.

This causal graph is used to design a scheduler where the scheduling policy is designed by taking into consideration the predicted burstiness in traffic and the priority of the tasks. Task priority is partially determined by the output of the causal graph. Appropriate scheduling policy is useful in avoiding the boot time of the virtual machines by starting them prior to their requirement in real-time. This reduces the latency which is bound to arise in the absence of such a prediction scheme for resource allocation, as it approximately takes several minutes to start a new instance on a VM on the cloud [20]. One may argue that scheduler can schedule tasks on virtual machines that are already running, but this would be constrained load balancing, thereby, defeating the purpose of using cloud's elasticity. However, care must be taken to reduce over-allocation of available resources rendering them to be under-utilized. A feedback to the causal graph also exists from the cloud

to the causal graph to input how many more tasks a VM can handle. This information is then determined by the causal graph based on available buffer space and memory and used in outputting snapshots in *Level 3*. Based on the feedback from the cloud, the most appropriate queuing model that gives a locally optimum scheduling output is chosen to reduce the wait times in the queue and the server, thus, keeping the latency always in the bounds. The algorithms in Level 1 and 2 are iteratively run to obtain locally optimum latency values over several time windows, as the globally optimum values cannot be achieved as latency is dependent upon decision making process which is an NP-Complete problem.

- Level 3 pertains to the analysis and distribution of the information onto the cloud servers and data dissemination back into the control loop.

In figure 1, the components of level 3 are the system snapshots (s_i, s_j, s_k, s_l, s_m), the decision maker D and the cloud. Scheduling tasks on the cloud servers (VMs) is based on task priority and QoS requirements of the CPS. Tasks in the CPS application used in this paper basically correspond to the processing requirements based on the area of observation (see sec. VI). Task prioritization refers to the importance of the processing of one set of data over the other based on the impact it has on the overall stability and operability of the CPS. Task/content distribution based on prioritization is done to maximize utilization for efficient information processing. Efficient scheduling enables quick re-transfer of data back to the control unit with locally optimal latency as it improves the processing on the VMs by reducing the wait times. Re-transferred data also contains information in control messages that is sent to the sensors, which governs the new set of sensor readings to be taken, for better information processing at the next time instant. The decisions taken by the fusion architecture with the aid of this new information improves the overall system behavior. The information fed to the fusion architecture and control decision feedback forms the communication loop and satisfies the control requirements of the physical component of the CPS and closes the control loop.

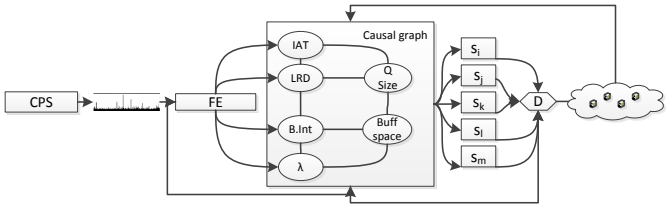

Figure 1. Information Fusion Architecture

B. DETAILED FUSION ARCHITECTURE

1. Level 1: Feature Selection.

Feature selection in the information fusion architecture is a method by which a subset of predominantly discriminative features are selected among the entire set of features characterizing CPS. This is an important pre-screening and filtering process which allows the user to focus information processing on important and defining characteristics of the traffic. In our fusion architecture, we use feature extraction in a manner which aids us with determining the network characteristics of the traffic trace. Parametric information such as the degree of self-similarity, arrival rates, service rates, network delays, system noise etc. are extracted to help obtain multiple snapshots of the system representing system behavior under changing parameters in the next level. After this, correlations among the chosen parameters are computed. This is done in order to help build causal relations among the parameters to update the decision pool with information about new network/system snapshots. Among the snapshots that are output using the features in the decision pool, the most appropriate snapshot which satisfies the QoS requirements of the CPS is selected. Furthermore, parameters chosen from the feature selection algorithm are useful in determining the probability of bursty intervals in the incoming traffic.

In the feature selection and update algorithm, let F be a pool of features to be se-

lected from. At the start of the fusion process, the pool is empty, as no features are selected for analysis. Later on, upon adding new features f_i and removing low impact features f_k (i.e. features whose extraction and use in further decision making process will not improve overall QoS), we obtain a new feature pool F_{new}. This is done by calculating the symmetric uncertainty among the attributes of the features. The correlation co-efficient γ is used as a threshold to determine the most appropriate features to be selected and input to the next level in generating a causal graph. The correlation co-efficient determines the strength and the overall effectiveness of the causal graph. It is empirically determined based on QoS requirements under the constraint of keeping the latency under the maximally allowable delay τ_{mad} as given in sec. B. γ is application specific and could change and is not necessarily preset. The algorithm is as follows:

Algorithm 4: LEVEL 1: Feature Selection and Update Process

Input: Incoming traffic stream, Information about feature dependence
Output: Inter-arrival time, Task segregation, Self-similarity, Queuing model, arrival rates
Initially $F = \Phi$
while $i < F$ **do**
  $\forall f_i \notin F$
  Perform $f_i \bigcup F$
  Update $F \rightarrow F_{new} = F \cup f_i$
  Update $F_{new} - f_k \; where, \; f_k \in \gamma$
  $*remove\ low\ impact\ features$
  return $F = F_{new}$
end
determine $task\ T\ \forall\ f_i \in F$

Feature selection has a greedy approach and sometimes might not give the desired output. In such a scenario, we propose to multiply the autocorrelations with random values in order to make the set of values more uniformly distributed. By doing so, we make sure that new features can also get added into existing features (parameters) in the feature

pool. This approach adds edge pruning to the fusion architecture and makes it more accommodating and fair; thus, making it more accurate. It must be noted that the feature selection algorithm chooses parameters that are network specific and not cloud specific and they are still very much applicable to large scale cloud deployments such as is the case with Amazon.

Time complexity analysis: Calculation of symmetric uncertainty can be done in linear time depending on the number of features k used from the traffic trace. Since the feature selection algorithm is iterative and at each instant at an average half of the features in the pool are eliminated. This gives us a time complexity of $O(NlogN)$. Hence, the overall time complexity is $O(kNlogN)$.

2. Probabilistic Burst Prediction and traffic analysis.

Traffic packets arrive at different times. They may arrive in a burst or after long silence between bursts. For example, n number of packets may arrive in x seconds, or, there is x seconds of silence between the arrival of two packets. This shows that when packets come in bursts, it is very likely that arrival of the next packet will also be as part of the burst. Also, the next packet that arrives following packets that arrive with long inter arrival silence is more likely to come in a burst. The first step in computing whether the next packet is bursty or non-bursty is to obtain information about the packets' arrival time. Network traffic input is obtained as Pareto Distribution with the restriction $1 < \alpha < 2$, where α is the shape of the distribution. This maintains the long-range dependence. τ is used to control the size of bursty interval and non-bursty interval. To compute the total number of packets in the interval (whether bursty or non-bursty), we calculated the probability of the type of the incoming packet i.e. if it belongs to the bursty or non-bursty interval.

$$P[X]_{non-bursty-packet} = 1/(1+\sum T_{p\ bursty-interval}), \quad when\ t_{arrival} > \tau \tag{8}$$

$$P[X]_{bursty-packet} = 1/(1+\sum T_{p\ non-bursty-interval}), \quad when\ t_{arrival} \leq \tau \tag{9}$$

$$P_{average\ of\ non-bursty-packet} = 1/n\sum P[X]_{n\ non-bursty-packet}, \quad where\ n = 1,2,3,...n \tag{10}$$

$$P_{average\ of\ bursty-packet} = 1/n\sum P[X]_{n\ bursty-packet}, \quad where\ n = 1,2,3,...n \tag{11}$$

where, $P[X]$ is the probability that the next packet arrival time is greater or less than τ and T_p is the total number of packets in the interval. Once we have achieved this, we compute the probability of the transition of the entire transaction up to time t by computing the average of the probability. Once these probability values are obtained, the time interval over which the bursty packets are predicted is noted.

3. Level 2: Causal graphical model construction. In this level, we build a causal graph by building causal relations among the parameters. Causal relations help us understand the effects of a change in one network parameter with respect to the other and are built based on the correlations between the network parameters. The network is dynamic in nature, i.e., causality changes over a period of time based on: changing QoS requirements, number of dependent tasks, network behavior etc. In order to capture this dynamicity, appropriate correlation co-efficient has to be chosen in a manner which gives locally optimum values from the decisions output at this level, and, which satisfy QoS while ensuring CPS stability through bounded latency. Thus, the choice of the threshold δ is of utmost impor-

tance and must be computed periodically over several time windows based on the prediction and determination of the traffic arrival. The causal graph has nodes which are made up of network parameters like the arrival rates, waiting times, queue lengths and queue sizes, Hurst parameter values, shaping function α, service times etc. The most important criterion to be satisfied in CPS is latency reduction while maintaining QoS, i.e. lower latency condition which does not satisfy required QoS will not be chosen. Such a reduction in latency will not show performance improvements. Thus, in order to determine the locally optimal latency values, we propose the idea of outputting several network *snapshots* of the future system state and choosing from among the most appropriate snapshots that reduces overall latency and better represents the system behavior. This is a quicker way of pro actively ascertaining decisions as opposed to reactive iterative algorithms used in decision making.

4. Level 3:Snapshots and decision making. Snapshots $s_i \in \{S\}$ contain the calculations of the service times in relation to the most important parameters that form the feature pool. Service times reflect the wait times in the queues, delays over communications, task re-prioritization and resource re-allocation and task re-scheduling. This is done over several time intervals by sliding the time window as new traffic arrives. Using the snapshots, we can make decisions to preempt the Cloud to scale its resources accordingly.

C. TASK SCHEDULING ALGORITHM

We would now like to describe our task scheduling algorithm that takes into account the above mentioned QoS constraints on task priority. Task scheduling on the VMs is extremely critical as it determines the overall makespan. It is also instrumental in meeting with the application specific delay constraints. Our Task Scheduling Algorithm (TSA) starts by prioritizing the tasks based on the importance of their processing on the overall system behavior. Our scheme, firstly prioritizes the tasks based on their importance and time-criticality, then, calculates the mean service time on a virtual machine on the cloud, the average waiting time in the queue and schedules the requests or the incoming traffic

Algorithm 5: LEVEL 2: Information fusion architecture for scheduling under variable loads

```
Input: Features F from level 1, parameter correlations
Output: System snapshots, Control Messages, Task scheduling, Task
        prioritization, Dependence graph G
for ∀{f_i, f_j} ∈ F do
    generate dependence graph G
    while G do
        compute γ ∀ {f_i, f_j}
        compute edge weights w for dependence graph
        {G}
        w = task_priority{f_i, f_j} + γ{f_i, f_j}
        generate snapshot{S}
        compare {s_i, s_j} ∈ {S}
        {G'} = sorted {G}
    end
    update {G} with changed w
    determine snapshot ← current{G}
    ***SCHEDULING {Sc} ON VMs ***
    determine available space on VMs, W_q
    calculate queue length(Q), memory requirement
     on VM, τ_mati, τ_mad
    update snapshot{S}
    allocate tasks t_i ∈ {T} to VMs ; Schedule{Sc}
    determine completed tasks t_k
    update Schedule{Sc'} = {Sc} − t_k
    generate control messages
    return Snapshot, Control Messages, Sc
    information, τ_mati, τ_mad, Task completion time
    END
end
```

into the queues accordingly for each of the parameters. This ensures that resources are provided to the tasks with high urgency.

Scheduling strategies proposed in [24], pertain to scheduling a VM to perform a task; bringing up new instances based on the needs of incoming tasks has not been handled effectively. Also, in schemes such as [24], scheduling corresponds to the load balancing of files on the VMs; instead of prioritizing tasks and performing resource allocation for a

real-time task-critical application. Our queuing based fusion architecture takes appropriate decisions to overcome this task scheduling issue. Cloud platforms using EUCALYPTUS use regular scheduling schemes such as *Rand* or *Qlen* are used. Our proposed scheme improves on them in scheduling tasks based on the QoS requirements. Using the concepts of queuing theory, the scheduler calculates the mean service time on a VM on the cloud, the average waiting time for a schedule to be cleared off the queue (waiting time on the queue W_q) and schedules the incoming tasks accordingly. The tasks are prioritized based on γ and its out/in degree (dependency on other tasks). This ensures that resources are provided to the tasks with high urgency. However, there is a down side to this approach. There could be tasks that might never meet the emergency criteria and remain in the queue forever. In order to break such a deadlock, we define a preset maximum allowable time threshold ψ beyond which it has to be allotted a computing resource. ψ is determined using τ_{mati} - the maximally allowable time interval before another task on the queue is processed and τ_{mad} is the maximally allowable delay before which QoS requirements for the CPS have to be serviced. This threshold value is obtained by taking as reference, the largest allowable waiting time on a task based on previous experiences. Algorithm 2 describes our Task Scheduling Algorithm in its completeness.

1. Algorithmic Complexity. We now determine the time complexity of our scheduling algorithm which is based on a DAG. The complexity is computed based on the number of tasks t, number of edges e, and number of nodes n.

1. There are t tasks in the DAG for parallel computing. For each task t_i, the computation of task priority can be done in $O(n + deg)$, where deg is the maximum in degree or out degree of the task. Therefore, the overall time taken to compute the task priority is $O(t(n + deg))$.

2. After computing the task priority, a efficient sorting algorithm can be used to sort the tasks in logarithmic time $O(tlogt)$.

3. To determine the task completion time for all the tasks for all the nodes based on the QoS requirements, the complexity is $O(n * deg)$.

4. To allocate the tasks to all the resources, in the worst case scenario, the execution time is $O(n)$. In order to determine the overall scheduling time we have to consider the completion time and the allocation time together. Hence, the time complexity is $Ot(n * deg + n)$.

5. The overall time complexity of the entire algorithm is $O(t(n * deg + n) + tlogt + t(n + deg))$.

VI. EXPERIMENTAL SETUP AND PERFORMANCE ANALYSIS

We performed experiments on real data from a CPS application called the ICE-L experiment [25] which captured the Cloud Condensation Nuclei (CCN). CCN is useful in predicting the rainfall and is in turn used in weather predictions. The data contained supersaturation values below which the particle concentrations would activate into cloud droplets. The total data set has 1.5 million data points from different data sets of the same experiment. We then used slice interpolation on the real data twice to generate 30 and 100 million data points. Additional synthetic data was generated owing to lack of readily available real big data in CPS. We can assure that interpolated values were statistically consistent with the real data. We used Amazon's EC2 instances to run our experiments. We used 4 small instances 3 medium instances and 2 large instance with RAM sizes 4Gbytes, 8Gbytes and 16Gbytes and with 4 cores, 8 cores and 16 cores. We compared our results to two standard schedulers as provided by Amazon, a Hybrid scheduler as devised in [4], Critical Path Genetic Algorithm (CPGA) [14] and the Duplication Scheduling Heuristic Algorithm (DSH) [27]. From our experiments on Amazon, we largely observed that very large network delay is rare. However at times, the delay was larger than desired and the unexpected delay is incorporated into the determination of upper bound of latency in our architecture.

A. SELF-SIMILARITY PREDICTION

We ran our probabilistic burst prediction algorithm to predict the burstiness of the self-similar CPS traffic (CCN traffic) and the CCN dataset. As we can see from Figure 2, the faint blue bursts is the actual CCN data and the red data is our estimation. Self-similarity can be seen in both data and the traffic. In order to study the effects of long range dependence and self-similarity, traffic with varying self-similarity and traffic burstiness

was used and the effects of varying queue sizes and buffer spaces in the VMs' memory was studied. By implementing our algorithm, we captured the self-similar behavior of the traffic and predicted the traffic behavior. From Figure 2, in instances where the spikes (bursts) were larger than the predicted amount of traffic, we noticed a negligible change in the performance of our algorithm. This is because our fusion architecture selects the queue size based on the expected incoming traffic and sets the queue limit to a little more than the expected value. This queue size adjustment is done to ensure that any unexpected burst is accounted for and served appropriately. Accuracy in predicting the traffic traces were effective in determining the LRD of the traffic and estimating it. It helped us in choosing our decisions about the queue lengths and the appropriate load distribution solution accurately over different time windows. Figure 3 shows a sample of the bursty packets in the traffic trace. It gives us an understanding of the peak bursts and average bursts that is being processed by our scheduling algorithm.

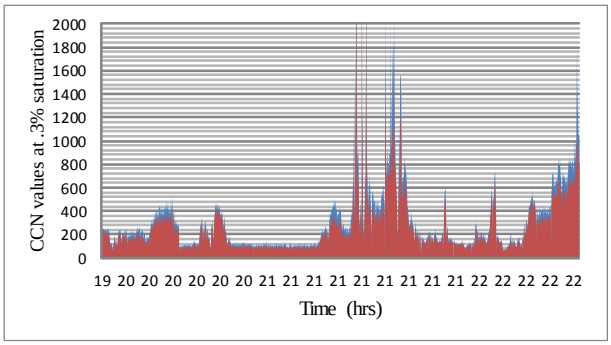

Figure 2. Self-similarity in the CPS data

B. EFFECTS OF QUEUE LENGTH ON THE TAIL AND VICE VERSA

We studied the effect of queue lengths on the tail distribution to study the effects of

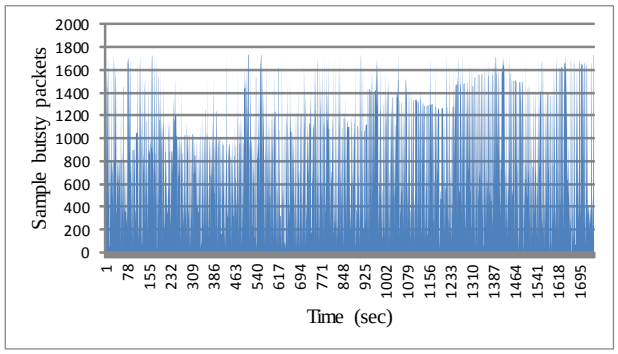

Figure 3. Bursty workload of the CPS traffic

LRD on the decisions taken by our architecture. Our simulation results both indicate that given a very small queue size Q, as Hurst parameter H increases from 0.59 to 0.78, tail distribution (probability of possible long range dependence) decreases, but not by much. This happens because if there are small buffer spaces, then resetting and truncating effects are bound to show up. When the queue is reset or emptied in a small buffer size, the correlation degree of adjacent-input traffic becomes weak and effect of LRD can be studied only when traffic streams accumulate in buffer in comparable quantity. Meanwhile, messages will be discarded if there is a buffer overflow, which further weakens the effects of LRD traffic. Figure 4 displays a trend where, the tail distribution as well as the queue size are inversely proportional to an increase in the self-similar nature of the traffic as indicated by an increase in the Hurst parameter. This trend is observed as our Information Fusion Architecture (IFA) increases the buffer space to hold more packets thereby, reducing the probability of long-range dependent data outside the queue. This is done in order to efficiently re-schedule the tasks on the VMs.

This trend is observed because there is no resetting and truncating with a large buffer size and therefore, the level of relevance of long range dependent traffic becomes more significant. As a consequence, with an increase in the Hurst parameter, larger buffer sizes

are chosen so as to smooth traffic flows in the queuing system. As the effects of LRD traffic become more obvious, longer queue length (thereby, bigger queue size) and poorer queuing performance is observed. A poorer queuing performance can be related to an under utilization of the queue when there are only intermittent bursts, not continuous. Thus, knowledge about the effects of Hurst parameter on queue size is of high significance in choosing the queue lengths. Assuming we have an infinite queue, study of the effect of queue length vs Hurst parameter is not of any great significance. But, even with the use of cloud computing, where a server can be dedicated to holding and maintaining queues, in most practical applications, an infinite queue length is still a far fetched assumption. Hence, using queuing theories based on an infinite queue length has its limitations in real-time scenarios. However, in the case of a CPS applications, because of the nature of the traffic and the traffic arrival rates, a finitely infinite buffer on a cloud platform is a safe assumption. The information obtained from these results is fed into the decision making process as described in section 3 to enable greater efficiency in coming up with a decision that could predict the snapshot at a future time with respect to the choice of the queue size, wait times and buffer space available (which is used in determining the VM allocation).

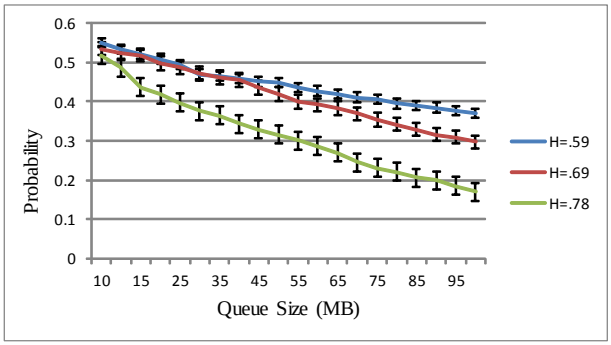

Figure 4. Varying tail probability with change in queue size

C. UTILIZATION AND SPEEDUP

Utilization is defined as the ratio of the capacity of the VM that is in use at a given instant of time over the total VM processing capacity. It can also be interpreted as the time for which a VM is busy over a given time period. Utilization $\rho = \frac{\lambda}{\mu * C}$ is a measure of how effectively the scheduling algorithm performs. We conducted experiments to determine utilization with respect to the queue sizes and the total load on an instance over a time period. Figure 5 shows how the percentage utilization varies for different instances of VMS. Figure 5 represents the utilization of each of the small, medium and large instances for the three different data sets with varying burstiness. We can observe that the utilization of large instances in the real data set containing 1.5 million nodes is low. This is because our algorithm determined that usage of small and medium instances was sufficient to process tasks. As we can see, the large instance was only used for 38% of the time, thereby reducing unnecessary resource provision. Large instances were only used for tasks with higher priority and in cases where the wait times in the queues would result in long latency values. Similarly, in figure 5, we can see that in the cases of 30 million and 100 million data points, the utilization of medium and large instances was high. This is determined by our algorithm based on the size of data to be processed, the increase in task dependency and the corresponding maximally allowable delay that is computed. We see a reverse in the utilization trend for the larger data sizes because our algorithm now uses the small instances to solve processing contention and congestion in the queue and additional parallel processing required to obtain quicker latency values.

We considered speedup as another performance metric to study the effectiveness of our scheduler in task prioritization and resource allocation. Speedup is estimated as $S = T(1)/T(C)$ i.e. the time taken to execute a set of tasks on a uni-processing unit vs the time taken to perform same set of tasks using C processing units (VMs). We compared our

results with Critical Path Genetic Algorithm (CPGA) [14] and the Duplication Scheduling Heuristic Algorithm (DSH) [27]. As we can see from figure 2, our algorithm outperformed the other two algorithms except in the case of 4VMs for CPGA. This is because of the availability of extra processing which aids the genetic algorithm in [14] at that particular time instant. However, this makes [14] static as any reduction in the VMs will affect their performance as opposed to our algorithm which can adjust to fluctuations in the incoming data to be processed. Hence, our algorithm scales smoothly over variable number of VMs. Our algorithm showed a linear increase in speedup despite bottlenecks such as bandwidth constraints, whereas, the other two algorithms performed sub-linearly. This shows that the snapshots generated by our scheduler outputs decisions upon considering the over all performance increase, instead of just considering latency reduction.

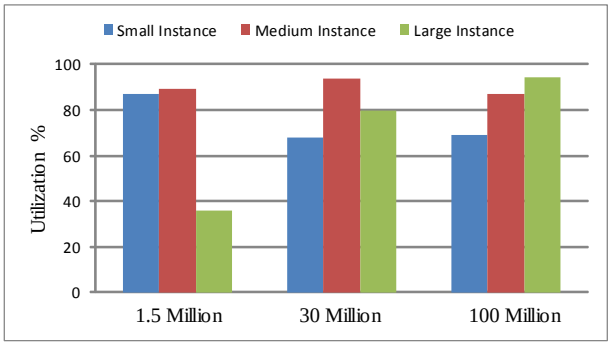

Figure 5. Utilization of instances

D. EFFECT OF QUEUE LENGTH ON BUFFER SIZE IN MEMORY AND PACKET LOSS

In our analysis of the throughput and packet loss rate, the improvement in performance for large buffer sizes was gradual. Figure 7 shows that the qualitative dependence

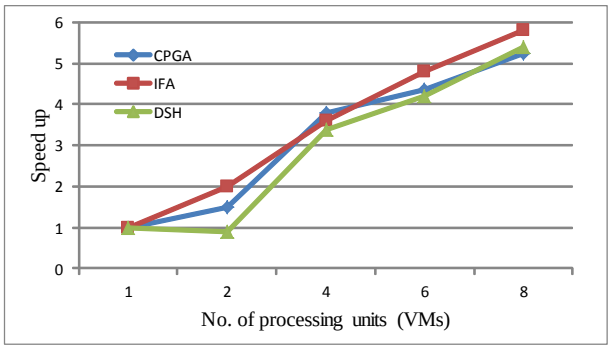

Figure 6. Speedup vs Resource allocation

of mean queue length for large buffer sizes under packet loss is determined by the degree of self-similarity. For H close to 0.5, the functional dependence of mean queue length on buffer capacity is clearly sub-linear (roughly logarithmic) whereas for H close to 1, the dependence becomes linear. For very large values of the buffer size we expect the delay curve to saturate even when α (shaping parameter of the distribution) is close to 1 although the practical significance of the difference in the magnitude of their queuing delay will be preserved; because, when the shape parameter is close to 1, the distribution has infinite variance and finite mean and the queue size decays slowly in a hyperbolic fashion indicating the presence of long-range dependence. The proportional increase in buffer occupancy for H close to 1 indicates that even for large buffer sizes, packet drops included in traffic burstiness is high enough to cause constant utilization to meet the requirements of unserviced tasks. Figure 7 represents a difference in the growth of the queue size upon using IFA as compared to using Amazon's inbuilt scheduler for various H values. As we can see, even with an increase in packet loss %, our architecture performs well as snapshots are generated by varying the parameters which is representative of the network performance and CPSs QoS requirements. A further increase in packet loss will only see our algorithm showing a sublinear increase due to its nature of dynamically adapting to longer task exe-

cution times by resource allocation and queue management. However, our algorithm will still be excuted in superlinear time despite the sub-linear increase in queue size. Otherwise, a linear increase of queue size can manifest an exponential growth in the time complexity due to inaccurate algorithmic approximation.

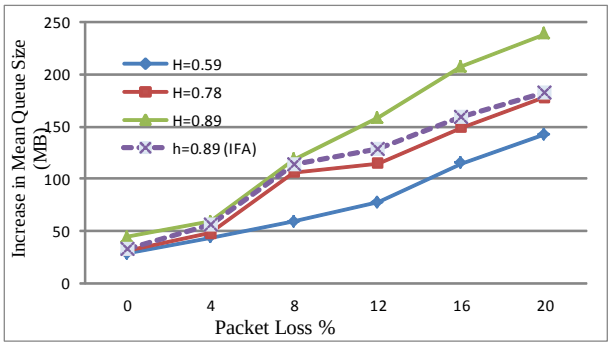

Figure 7. Effect of queue length on packet loss percentage

E. LATENCY

As stated in the section II, the main aim of this paper is to have bounded latency at all times to enable fast data dissemination back from the cloud. Our results in Table 1 prove the efficiency and the effectiveness of our approach in bounding latency as compared to other scheduling approaches. Our experimental data comprises of multiple data sets with differing QoS and task processing requirements, which refutes any claim that partial heuristic nature of our algorithm conforms to a certain type of data. From the complexity analysis in sec. 1, we can see that the overall processing latency is bounded by the number of features f, number of tasks t, the number of nodes n and their degree deg. Based on that, the worst case complexity was seen as task dependency grew with the number of tasks t. However, since independent tasks also existed, we observed a reduction in the time

complexity to finish a batch of tasks. Our IFA makes decisions about bringing up new resources dynamically based on both the completion time of the service and the wait times in the queues. Hence, our entire IFA works seamlessly under variable traffic conditions based on the feedback from each level of its implementation.

Table 1 shows the minimum and maximum latency values for each of the three different data sizes with varying burstiness from low to high. The delay τ_{mad} before which cloud should be able to send back decisions for 1.5 million and 30 million data sizes was 5 seconds, whereas, for 100 million points the allowable delay was 10 seconds. This delay is calculated based on how data is pooled from the sensors. We observed that the total latency time even under bursts is always bounded in our Information Fusion Architecture (IFA) as compared to the Hybrid 1 scheduling algorithm in [4] and the $RoundRobin(RR)$ and $Qlen$ scheduling used by Amazon. The latency times as shown in Table 1 are representative of the minimum and maximum of the sub-optimal latency times over different time periods. We say latency is sub-optimal because under variable traffic, obtaining the optimal solution is not possible. Latency includes the network delays as observed upon migrating the data to the cloud. Amazon's Elastic Load Balancer dedicates the VMs prior to running any experiment and uses EUCALYPTUS's inbuilt scheduler which is static and cannot handle burstiness effectively. However, upon use of our architecture on Amazon cloud, we have seen a significant reduction in the latency as compared to other schemes. Hybrid 1 scheduling algorithm in the table is a decentralized scheme which is used to process varying bursty loads, was run under the same constraints as our algorithm for fairness of comparison. However, Hybrid 1 does not take into consideration dynamically changing traffic burstiness and the scheduling is preset based on burstiness predicted at the start of the simulation. *RR* and *QLen* are the more common schemes used in EUCALYPTUS cloud and are not suited to dynamic changes in traffic traces and hence cannot perform well under bursty traffic conditions. The latency times shown in the table also reflect the decision making efficiency of the algorithm at several time instants. The results are representative

of how the scheduler makes use of the resources that are both available and in reserve. As we can see, latency values for varying burstiness for three different data sizes in IFA are lower and always bounded owing to the nature of our algorithm which is both iterative and dynamic in determining the queue sizes, buffer spaces and in determining the arrival of incoming traffic stream.

Table 1. Overall latency for CPS data processing

Data	IFA	Hybrid 1	RR	Qlen
1.5 Million	.52 - 2.10sec	.61 - 4.81sec	3.47 - 8.36sec	3.78 - 14.23sec
30 Million	1.20 - 3.89sec	2.91 - 5.89sec	5.48 - 15.57sec	7.34 - 14.21sec
100 Million	1.6 - 6.3sec	3.85 - 11.45sec	7.89 - 34.36sec	6.57 - 41.23sec

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a causality based information fusion architecture to bridge the gap that exists between scalable processing with minimal latency for time-critical information dissemination in a CPS. We have studied the effects of self-similarity and long range dependence in predicting the traffic bursts over time intervals. We have then devised an information fusion architecture that effectively manages task scheduling using load balancing on the cloud. We learned that even with an adaptive scheme, it is difficult to achieve near 100% utilization on instances under bursty and non-deterministic traffic. We determined that latency is lower bounded by network and traffic characteristics, and, over any time period if the latency is below the theoretical upper bound under those constraints, incremental reduction in latency does not improve the overall performance. The extensive simulations we conducted and results and analysis from them validate this claim by observing how different network analysis parameters like tail probability, queue length etc. affect scheduling on the cloud. In future, we would like to extend our work to see how inclusion of security breaches in the cyber and/or physical sub-system(s) will affect scheduling under variable loads and bursts for a cloud-assisted CPS.

REFERENCES

[1] A. B. Sharma, F. Ivančić, A. Niculescu-Mizil, H. Chen, and G. Jiang, "Modeling and analytics for cyber-physical systems in the age of big data," *SIGMETRICS Perform. Eval. Rev.*, vol. 41, no. 4, 2014.

[2] P. Bogdan and R. Marculescu, "Workload characterization and its impact on multicore platform design," in *Hardware/Software Codesign and System Synthesis (CODES+ISSS), IEEE/ACM/IFIP International Conference on*, 2010, pp. 231–240.

[3] Y. Yazir, C. Matthews, R. Farahbod, S. Neville, A. Guitouni, S. Ganti, and Y. Coady, "Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis," in *Cloud Computing (CLOUD), IEEE 3rd International Conference on*, 2010, pp. 91–98.

[4] J. Zhang, N. Mi, J. Tai, and W. Meleis, "Decentralized scheduling of bursty workload on computing grids," in *Communications (ICC), IEEE International Conference on*, 2011, pp. 1–5.

[5] L. Lu, P. Varman, and K. Doshi, "Decomposing workload bursts for efficient storage resource management," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 22, no. 5, pp. 860–873, 2011.

[6] N. Mi, G. Casale, and E. Smirni, "Aside: Using autocorrelation-based size estimation for scheduling bursty workloads," *Network and Service Management, IEEE Transactions on*, vol. 9, no. 2, pp. 198–212, 2012.

[7] Y. Zhao and W. Huang, "Adaptive distributed load balancing algorithm based on live migration of virtual machines in cloud," in *INC, IMS and IDC, NCM. Fifth International Joint Conference on*, 2009, pp. 170–175.

[8] M. Stillwell, D. Schanzenbach, F. Vivien, and H. Casanova, "Resource allocation algorithms for virtualized service hosting platforms," *J. Parallel Distrib. Comput.*, vol. 70, no. 9, September 2010.

[9] J. Liebeherr, A. Burchard, and F. Ciucu, "Delay bounds in communication networks with heavy-tailed and self-similar traffic," *Information Theory, IEEE Transactions on*, vol. 58, no. 2, pp. 1010–1024, 2012.

[10] M. J. Khabbaz, W. F. Fawaz, and C. M. Assi, "A simple free-flow traffic model for vehicular intermittently connected networks," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 3, pp. 1312–1326, 2012.

[11] Z. Wang, K. Yang, and D. K. Hunter, "Modelling and analysis of multi-sink wireless

sensor networks using queuing theory," in *Computer Science and Electronic Engineering Conference (CEEC), 2012 4th*. IEEE, 2012, pp. 169–174.

[12] M. Hedayati, S. H. Kamali, and A. S. Izadi, "The monitoring of the network traffic based on queuing theory and simulation in heterogeneous network environment," in *Information and Multimedia Technology, 2009. ICIMT'09. International Conference on*. IEEE, 2009, pp. 396–402.

[13] F. Li, D. Yue, and C. Li, "A query processing approach based on queuing model for cyber-physical systems," *Conference, International Asia-Pacific Web*, vol. 0, pp. 291–297, 2010.

[14] Q. GAO and G.-x. LI, "A traffic prediction method based on ann and adaptive template matching," *This paper has been submitted to Przegld Elektrotechniczny with paper ID: PE2257*, 2011.

[15] F. Wang, D. Li, and Y. Zhao, "Prediction of self-similar traffic and its application in network bandwidth allocation," in *Wireless Communications, Networking and Mobile Computing (WiCom). International Conference on*, 2007, pp. 1980–1983.

[16] G. Casale, N. Mi, L. Cherkasova, and E. Smirni, "Dealing with burstiness in multi-tier applications: Models and their parameterization," *Software Engineering, IEEE Transactions on*, vol. 38, no. 5, pp. 1040–1053, 2012.

[17] J. Cardoso, P. Derler, J. Eidson, and E. Lee, "Network latency and packet delay variation in cyber-physical systems," in *Network Science Workshop (NSW), 2011 IEEE*, 2011, pp. 51–58.

[18] H. Li, "Virtual queue based distributed data traffic scheduling for cyber physical systems with application in smart grid," in *Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on*, 2012, pp. 309–314.

[19] M. Levorato and U. Mitra, "Scale invariance and long-range dependence in smart energy grids," in *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), Asia-Pacific*, 2012, pp. 1–8.

[20] M. Olson and K. Chandy, "Performance issues in cloud computing for cyber-physical applications," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, 2011, pp. 742–743.

[21] K. Li, G. Xu, G. Zhao, Y. Dong, and D. Wang, "Cloud task scheduling based on load balancing ant colony optimization," in *Chinagrid Conference (ChinaGrid), 2011 Sixth Annual*, 2011, pp. 3–9.

[22] K. Park and W. Willinger, *Self-Similar Network Traffic and Performance Evaluation*,

1st ed. New York, NY, USA: John Wiley & Sons, Inc., 2000.

[23] X. Jin, G. Min, and S. Velentzas, "An analytical queuing model for long range dependent arrivals and variable service capacity," in *Communications, 2008. ICC. IEEE International Conference on*, 2008.

[24] K. Ye, X. Jiang, D. Huang, J. Chen, and B. Wang, "Live migration of multiple virtual machines with resource reservation in cloud computing environments," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*. IEEE, 2011, pp. 267–274.

[25] http://data.eol.ucar.edu/codiac/projs?ICE-L.

[26] F. A. Omara and M. M. Arafa, "Genetic algorithms for task scheduling problem," *Journal of Parallel and Distributed Computing*, vol. 70, no. 1, pp. 13 – 22, 2010.

[27] M. Wosko, I. Moser, and K. Mansour, "Scheduling for optimal response times in queues of stochastic workflows," in *AI 2013: Advances in Artificial Intelligence*, 2013, vol. 8272, pp. 502–513.

IV. QoS GUARANTEEING ROBUST SCHEDULING IN ATTACK RESILIENT CLOUD ASSISTED CYBER PHYSICAL SYSTEMS

Brijesh Kashyap Chejerla, Sanjay K Madria

Department of Computer Science,

Missouri University of Science and Technology, Rolla, Missouri 65401

In this paper, we propose a new approach to the semi-network form game theory to obtain a robust and attack resilient scheduling mechanism for a Cyber Physical System (CPS) connected to a cloud computing platform. The need for a cloud computing platform arises owing to the ever increasing processing demands of CPS applications, which are becoming increasingly heterogeneous. In future, this heterogeneity will demand big data processing of CPS applications. However, in time and mission critical CPS applications, big data processing has to satisfy stringent Quality of Service (QoS) requirements to seamlessly close the control and decision feedback loop. Such QoS requirements will need a scheduling mechanism to efficiently utilize the scalable processing components as provided by a cloud computing platform. However, owing to the increase in the attack space of an attacker with an increase in access points, the scheduling mechanism needs to be robust even under security attacks. To address this issue, we develop an extension of the semi-network form game to aid the scheduling algorithm proposed in the paper. We conducted experiments on Amazon EC2 cloud and analyzed the results of our work in comparison with other existing scheduling mechanisms.

I. INTRODUCTION

Real world applications of Cyber Physical Systems (CPS) extend to various domains such as but not limited to *Transportation CPS and Supervisory Control and Data Acquisition (SCADA) systems* such as *Smart Grids*. The requirements and Quality of Service (QoS) parameters for each of the applications vary in nature (static or dynamic) based on the topological setting and the communication environment. The primary aim of the cyber component aiding such CPSs should be to maintain system operational stability and also to ensure security. In Transportation CPS, the job of the cyber component also extends to dynamically updating the overall system in lieu of new sub-systems and components being added on-the-fly, thus making it more complex. Therefore, in such dynamic hybrid systems, QoS requirements such as latency constraints of accurate processing and communication and feedback of control and decision are very stringent owing to stability concerns. Satisfying the varying QoS requirements of connected components while keeping the latency low to run the CPS smoothly is a non-trivial task and cannot be efficiently performed using only the existing infrastructure of processing components on the various CPSs mentioned.

Providing CPS security is very important in maintaining the stability of the system by satisfying the QoS requirements of the components. An attacker can cripple the system by attacking either the cyber or the physical component and cripple either one or both of them. In our paper, we discuss a game theoretic solution which considers the effects of attacks on either components that propagate to the other component. This task of keeping a hybrid CPS stable under various dynamism and security attacks necessitates the use of a Cloud Computing (CC) platform as the cyber component. CC creates more processing resources to cater to the increased computational needs of an unsecured CPS. In contrast to a centralized architecture of traditional CPS, cloud platforms provide a scalable and distributed

and heterogeneous platform with lower operational and setup costs. However, integrating cloud with a CPS comes with some design challenges, e.g. migrating computations onto the cloud could result in QoS constraints such as an increase in the overall information processing latency. Also, additional bottlenecks such as bandwidth constraints, resource unavailability etc. could arise because of the poor co-design of the cyber (CC) and the physical components.

Game theoretic principles pit multiple players with varying objectives against each other. In contrast to a pure optimization technique, game theory allows behavioral modeling of the attacker and defender over progressing time intervals. Each player has a set of tuples that govern the strength of his decision making abilities. The tuples consist of information about players' QoS requirements, players' observable system state and access points to the system. As games are played iteratively, employing a solution scheme that unrolls over multiple time slices reflects the system state at each iteration as accurate as possible. In order to deal with this problem, we delved into the domain of dependency graphs. Dynamic Bayesian Networks (DBNs) are a subset of dependency graphs that we have used in unison with the game theoretic framework. A similar concept was developed by authors in [11] using bayes-net and is called as Semi Network-form Game (SNFG) of human strategic behavior. However, our approach is different from existing work in that we use a time-varying DBN where causality among the random variables over multiple time slices exists inherently. This is more suited to a game theoretic framework, where decisions could be made with partial or incomplete knowledge of the opponents' strategy. Our DBN aided game theoretic framework produces control decisions and information that is useful for scheduling on the resources

Based on the outcome of the game over several time slice, decisions are taken which provide inputs to the scheduler. Task scheduling for a hybrid and possibly dynamic cyber physical system is a very complex process. Owing to the real-time latency constraints imposed on hybrid CPS by security attacks, this problem becomes more complicated as given

tasks have to be re-prioritized and rescheduled under the influence of an attack. Therefore, in this paper, we device a scheduling algorithm that builds on the information about the system state as output by the game theoretic approach. The task scheduling algorithm proposed in this paper is an effective solution for dependent / precedence-constrained tasks. Task dependency has to be taken into consideration as dependent tasks determine the on-time fulfillment of QoS requirements.

Our main contribution of this paper is to couple the cloud platform and the physical system by eliminating the various coupling constraints mentioned so far. Therefore, in order to eliminate coupling constraints and provide robustness, an extension of the semi-network form game and a QoS satisfying scheduling algorithm made robust by the semi-network form game have been proposed in this paper. This multi-layered solution is to provide a security-driven scheduler which can provide the required robustness. Such a framework should address the issues of attack modeling, response and action optimization and subsequently allocate resources to provide appropriate defense against security attacks.

In this paper, we provide details on the use of a SNFG and how we use a time varying DBN to capture the interactions of the game over multiple time slices. Next, we describe the security considerations upon which our scheduling algorithm is based. Finally, in our experimental evaluation, we will discuss how we used metrics such as *makespan, speedup, resource allocation and reliability* to determine the effectiveness of our proposed solutions in comparison with other techniques.

II. RELATED WORK

Scheduling algorithms are very crucial in obtaining high performances in heterogeneous computing systems such as cloud. The basic objective is to reduce the overall makespan. Directed Acyclic Graphs (DAGs) are used to represent the parallel computation problem and task scheduling, where nodes represent the tasks and edges represent the dependencies with other tasks. It has been shown that the problem of finding optimal schedule is NP-complete. The most common heuristic DAG scheduling is the traditional list scheduling algorithm. However, most list scheduling algorithms are designed for homogeneous systems [3, 2]. Several list scheduling algorithms have been proposed for heterogeneous systems, such as, mapping heuristic (MH) [16], dynamic-level scheduling (DLS) [21], Dynamic Critical Path (DCP) [1] and heterogeneous earliest-finish-time (HEFT) algorithm [3]. There is very little previous work in the field of security driven scheduling algorithms as most researches do not focus on that problem. Of the existing works on security driven scheduling, the QSMTS-IP [5] algorithm is capable of meeting diverse QoS requirements which includes security for multiple users, It focuses on minimizing the number of users whose tasks cannot be completed due to resource limitations. In [19], authors studied dynamic security-aware scheduling algorithms for single machine, homogeneous cluster, and heterogeneous distributed system. Their work can provide overall system schedulability under security. The closest work to this research is by [7, 18]. They developed three risk-resilient strategies and a genetic algorithm-based scheme, that is, the Space Time Genetic Algorithm (STGA), to provide security assurance in grid job scheduling. However, their algorithms cannot be applied in heterogeneous systems such as cloud. Unfortunately, these algorithms are applicable only on the cyber component and not in a bi-directional security scenario such as a cloud-assisted CPS.

For scheduling workflows, researchers have focused on a single QoS parameter -

makespan which makes them inefficient for CPS with many QoS parameters that affect the performance such as cost, reliability, and latency. The most popular approaches for scheduling are the static or dynamic list scheduling algorithms based on different heuristics such as opportunistic load balancing (OLB), minimum execution time (MET)[6], minimum completion time (MCT) [12], min-min, max-min [8] and heterogeneous earliest finish time (HEFT) [3]. Their basic idea is to determine an order of tasks based on heuristics and schedule the tasks accordingly. In addition to these heuristic techniques, there are also some meta-heuristic approaches such as genetic algorithms (GAs) [14], simulated annealing (SA) [9] and genetic simulated annealing (GSA) [25]. In general, meta-heuristic approaches manage to obtain much better performance, but take a longer execution time which can violate the basic condition of lower latency QoS requirement for a CPS application. In order to achieve this, we must make sure that the scheduler considers the trade-offs between different QoS parameters in order to satisfy the QoS requirements from both CPS and users and optimize the performance of the whole application. Buyya and Tham [24] proposed a workflow scheduling algorithm for economy-driven or market-driven computing platforms to minimize cost under the deadline constraint.

III. PROBLEM FORMULATION

In this paper, we address two specific problems that arise as a result of the coupling of a cloud platform to the physical system. The first problem is that of security and the second problem is that of task scheduling or more specifically workflow modeling under dependent tasks. Security is highly critical and difficult to achieve owing to the multiple access points for the attacker into the system. Thus, the security problem can be modeled as a game between the attacker and defender who have partial or no information about each others' strategy. In lieu of this complexity, we mathematically define the problem as follows :

A player of the game has to maximize his/her utility function U, while taking into account the costs of playing the game and opponents strategy. For a defender, the observation space Ω for his strategy reflects his knowledge about the system behavior as an operator. However, the defender will either have noisy or partial information about the system state under the possible presence of an attack. Similarly, an attacker will have incomplete information about the global system state. An attacker's control will be limited reflecting the part of the system which falls under his attack scope. Therefore, for both the defender and the attacker, the objective is:

$$\max U_D(x_{d_i}, c_d, Sc, \Omega_{n_d}) = E[{U_D}^+(x_{d_i}, c_d, Sc, n)] - C_D(x_{d_i}) \tag{1}$$
$$\max U_A(x_{a_i}, c_a, \Omega_{n_a}) = E[{U_A}^+(x_{a_i}, c_a, \Omega_{n_a})] - C_A(x_{a_i})$$

where, U_D, U_A is the utility function of defender and attacker, $x_{d_i}, c_d, Sc, \Omega_{n_d}$ are the best strategy for the system state of the defender at time slice i, cost of selecting the strategy, the scheduling function to be generated, and the observed space over a part of or all the nodes in the system. Similarly, $x_{a_i}, c_a, \Omega_{n_a}$ corresponds to the attacker's tuples. $E(\bullet)$ is the

expected utility of the attacker or the defender, where the players evaluate their strategy against the overall cost function $C_x = \{C_D, C_A\}$. Cost for a defender is computed from the increase in the number of resources required to maintain the reliability of the CPS. Similarly, an attacker's cost accounts to the requirements for an attack perpetuation with considerable gains.

Based on this understanding, we have proposed a solution to maximize the defender's utility function and force the minimization of the utility of the attacker to safeguard the system against security attacks.

The second problem that we solve in this paper is the selection of a scheduling design *Sc* for workflow among dependent tasks based on the input about QoS requirements and constraints from the solution to the afore mentioned first problem. The challenge is then to design a scheduler which effectively incorporates the requirements of the changing QoS such as latency, reliability and cost under security attacks.

IV. SECURITY USING GAME THEORETIC PRINCIPLES

A. SEMI NET-FORM GAME (SNGF)

In order to detect an attack and maximize the defender's utility function, we propose an extension of a Semi-Network form Game [11]. Semi-Network form games are a specific class of the network form games. It uses a Bayes Network consequently representing the underlying application as a probabilistic framework with random variables that are both observable and unknown. It combines the principles of game theory with time varying graphical structure to take actions even under no/partial information. In a semi-network game, the player can only control one decision node which captures his representation of the system state; whereas, in the normal game, players can control multiple decision nodes at the same time. Our work builds upon previous game-theoretic models of human-in-the-loop aircraft collision avoidance systems [11]. In this paper, we focus on developing the computational model for predicting the outcome of an attack on CPS where the operator is uncertain about the presence of an attacker. Further, we make extensions to the assumptions about limiting the operators' certainty that an attacker is present forcing the operator to perform well under both normal and attack conditions. We pro-actively predict the steps of further attack scenarios, thereby enabling us to design the physical and control system more effectively as opposed to a reactive approach. Thirdly, the extension to design requires numerical evaluation of many more scenarios, and in most scenarios, information about system state is not always available, either due to the presence of an attack or due to system faults. In order to counter this unavailability of information, we propose an architecture that fuses the principles of game theory and a time-varying DBN.

To summarize, we model and simulate the behavior of the defender and the cyber-physical attacker by developing reward functions and solution concepts using DBN that

closely represent the decision making processes of a human attacker or a machine. This differentiation is important, as the inferences a human can make about the system state with limited information and the subsequent decisions he makes is significantly different from those output by the utility function of an automated system/ machine that is used in an attack. Therefore, non-co-operative game models are embedded into the decision nodes of an SNFG that represents the evolution of the physical state and information available to both human decision nodes and the automation nodes. Choosing an accurate model allows the player to predict the outcomes of different system and parametric designs and consequently, maximize his own utility function. Such a design process finds similarities in the economic theory of mechanism design [24] where an external player designs a game under equilibrium conditions. The key difference between our approach and [24] design is that we do not assume equilibrium behavior which allows us to not settle to a solution for the control theoretic operations, but to explore all possibilities.

To predict how system design choices affect the outcome of attacker-defender interactions, we need a description of when player decisions are made and how these decisions affect the system state, i.e. a game definition. Sophisticated attacker strategies may be carried out over multiple time steps (i.e. many sequential and causal decisions), therefore, SNFG has to be expanded to fit such dynamicity. In order to satisfy this condition, a DBN is unrolled over sequential time slices to mimic the attacker-defender interaction. Each DBN is a causal graph $G = \{V, E\}$ with V nodes and E causal edges connecting time, and, has the structure of a distinct SNFG played out at time step i. These SNFGs are combined together to form an iterated SNFG by passing the system state s_i, the players' moves/decisions D_{D_i} and D_{A_i}, and the players' memories M_{D_i} and M_{A_i} from the SNFG at time step i to the SNFG at time step $i+1$. Since an SNFG has its components in both the cyber and the physical domain, we define the nodes that pertain to human behavior (attacker or defender) as decision nodes; and the nodes that are observed by the players in the game as chance nodes (the physical process). If a player plays SNFG against an automated machine with no in-

ferential capabilities, the strategies from such a node is still embedded as a decision node and not a chance node.

Formally, we define the SNFG as follows

Definition 1. An N-player semi-network form game is a quintuple (G, Y, U, R, π) where,

1. G is a directed acyclic graph $\{V, E\}$ with V vertices and E causal edges. The parent of a node is written as $par(v)$ and the child node is denoted as $chi(v)$, $v \in V$.

2. Y is the Cartesian product of $\|V\|$ with at least two elements.

3. U is a set of N utility functions $X \to \mathbb{R}^N$. Each player has his/her utility function which he/she looks to maximize.

4. R is a partition of the vertex set V into at least one chance node (physical system) and N player nodes or decision nodes.

5. π is the conditional probability distribution of each $v \in V$ that is conditioned upon its parent nodes.

In an SNFG, each player takes the best response action based on the observed information which is inferred from the system state. The statistical inference is performed by treating G, aided by conditional probability functions in π, as a Bayesian graphical model. It is difficult to derive a closed form of posterior probability distributions for a Bayesian network conditioned on some observed information. In order to alleviate this drawback, sampling techniques such as forward sampling, importance sampling or rejection sampling are widely used for approximating the posterior probability distributions. However, in SNFG, the major challenge is to sample from a decision node for which the conditional probability distributions are not specified. This is because π is only defined for chance nodes in the model.

We now describe the different nodes in the game. These nodes have causal interconnections based on which decisions are made by the player to maximize his/her utility

functions. At each time epoch, the nodes are causally interconnected and then unrolled into the next time step. Transition from one time instant to another time instant is based on the time-varying DBN.

1) Attacker existence: The decision to play a game against an attacker has to be made in order to ensure that optimal strategies are selected and the cost of selecting those strategies is minimized. Hence, there are two type of attacks that a game can be played against. One in which the presence of an attacker is detected and one in which there is unsurity about the presence of an attacker. With this in mind we define undetectable and unknown attacks as follows [15]:

Definition 2. An undetectable attack is defined as an attack, where, given initial conditions $y(x_1,u,t) = y(x_2,0,t)$, where, $\dot{x} = \{x_1,x_2\}, \dot{u} = \{u_1,u_2\}, t$ are the observed state variable of a measurement, the input to the system and the time instant, are input using an algorithm which is static; the observable output converges to the same value as is the case without any attack. It becomes completely undetectable if $t >$*attack signal* that starts after time 0.

Definition 3. An unidentifiable attack is defined as an attack where given initial conditions $y(x_1,u_1,t) = y(x_2,u_2,t)$ using an algorithm which is static, the observable output converges to the same value as is the case without any attack. The attack $a_i \subset A$, where, A is a set of stand-alone or collaborative attacks, becomes completely undetectable if $t >$*attack signal* that starts after time 0. An attack is unidentifiable if it cannot be distinguished from among an established set of attacks S.

Based on this, the probability of the presence of an attacker is given by p and this probability is used in obtaining the maximization of the utility function which selects the overall strategy. Varying values of p will determine the way the game is played, i.e. the defender strategies are changed and varied based on the attack probabilities.

2) System state: The nodes s_i represent the actual physical state of the cyber-physical system at the beginning of the time step i. In such cases, the defender's memory M_{D_i} and

the attackers memory M_{A_i} are held separately from the knowledge about the true physical state in order to incorporate the cases when either strategy or information about the players' observable space is unknown.

3) Observation Spaces: Observation spaces correspond to the information each player has about the system behavior and his/her control over the physical or cyber components given as Ω_{n_d} or Ω_{n_a} for the defender and attacker respectively. These observations are not complete (the players do not get full state information), they may be binned (indicating only the range of a variable, not the precise value), and also, they may be noisy. Here, Ω_{n_d} represents the visibility of the system as observed by the defender and can be assumed to be more consistent and informed owing to the nature of the defender being the CPS operator. However, it does not include information about the attack signal injected through the compromised node that has been taken over by the attacker. In contrast, Ω_{n_a} mostly provides information about the node captured from which, inferences can be made about the overall system state. Information from both Ω_{n_d} and Ω_{n_a} consists of the CPS' control state and design variables and would be inter-dependent on each other which further define the interactions among the defended and attacker in the game.

4) Player Memories: The content and evolution of player memories should be constructed using a DBN based on application specific information or experiments whose outcomes are decisions made by humans. This is important because decision bias incorporated by distinct humans gives a better understanding of the pseudo-optimal decision strategies that will arise while playing the SNFG.

The defender and the attacker memories are represented as $M_{D_i}, M_{A_i} \in M_i$ and consist of the observation space Ω_n, the players' moves/decisions $D_{D_{i-1}}$ and $D_{A_{i-1}}$ in the previous time instant and in the case of a defender knowledge of the presence of an attacker and system noise η. Therefore, $M_{D_i} = \{\Omega_{n_d}, D_{D_{i-1}}, p, \eta\}$. Similarly, $M_{A_i} = \{\Omega_{n_a}, D_{A_{i-1}}\}$. The memory of both the players is an exponentially decaying memory function of the form $M^i_{D_i} = (1-1/n)M^{i-1}_{D_i} + D_{D_i}$ for the defender and $M^i_{A_i} = (1-1/n)M^{i-1}_{A_i} + D_{A_i}$. This means

that the memory of a player will heavily be dependent on the number of time steps completed before a change in the decision is recorded.

4) Decisions and players' moves: The players decisions and moves are made by solution concepts such as Level-K reasoning as described in [11]. In this type of thinking model, the player makes the decisions based on the assumption that the opponent is at level k-1, which puts him perceivably at a better position.

B. TIME VARYING DYNAMIC BAYESIAN NETWORKS

In order to model the transitions among the iterative SNFG being played, we propose the use of a time varying dynamic bayesian network. Among several methods used for temporal and sequential analysis, dynamic Bayesian networks (DBNs) have been the most successful. A DBN is the extension of a Bayesian network (BN) in temporal domain in which conditional dependencies are modeled between random variables both within and across time slices. The conditional distributions are assumed to be homogeneous in DBN; that is, the structure $G[t]$ and parameter $\Theta[t]$ of DBN are fixed throughout the time. Under this assumption, a DBN is effectively constructed by unrolling a DBN in time axis, and the model learning procedure can be greatly simplified. However, this bold assumption limits the power of DBN in modeling many non-stationary sequences where the intrinsic relationships among variables change from time to time.

In this paper, we use a novel time varying dynamic Bayesian network (TVDBN) model for online inference. We extend the basic DBN model so that both the structure G and parameter Θ of network such as *attacks, delays* etc. become random variables that can change through time. These random variables $G[t]$ and $\Theta[t]$ are treated as additional hidden nodes in our graph model because they cannot be observed directly. Contrary to most off-line learning methods, we employ our feature extraction mechanism to dynamically infer the hidden states of network as well as missing data. Our framework is proposed in such a manner that any distribution, i.e. either multinomial or Gaussian is dealt with

effectively. A smooth transition prior probability is obtained which eases the problem of data scarcity. This novel representation of changing network allows a unified modeling of both data and network itself under the same dynamic Bayesian framework. In order to solve the problem of limited information to accurately obtain the conditional probabilities, we employ a feature extraction and selection method such as the particle filter. A particle filter (or Sequential Monte Carlo) is used to determine the state posteriori because it can handle arbitrary system and observation models. The posterior distribution is approximated by a finite set of state samples $\{s_t^i\}$ and its associated weights $\{w_t^i\}$.

$$p(s_t|o_{1:t} \approx \sum_{i=1}^{N_s} w_t^i \delta(s_t - s_t^i)) \tag{2}$$

where $o_{1:t}$ are the observed values up to time t. When N_s, the number of state samples (particles) approaches infinity, the approximation can be close to the true distribution. At each time epoch, filtering is done with the sample $\{s_{t-1}^i, w_{t-1}^i\}$ of the previous time. New samples are drawn from a proposal distribution $q(\bullet)$. With this, inference of the hidden states can be performed for different network structures and parameters with different dimensions.

We would like to mention the notable differences in our model and the ones that are comparable to it. A hidden variable is used to represent the change of network in [20], but it only serves as an auxiliary variable to facilitate implementation. The structure and parameter nodes in our TVDBN are principal components of the whole model, and they represent the statistical attributes of current network. The online adaptation methods in [13] can only model piecewise constant variation of network; while our method deals with both continuous change in parameter and discrete change in structure. Smooth change of network is ensured in [17] with a kernel window applied on data sequences; we achieve similar goal via a smooth transition model with the aid of feature extraction and parameter

selection for the generation of conditional probability, which is a more favorable solution from a Bayesian perspective. In the Gaussian graphical model [23], network parameter is marginalized out and network structure is the only thing that is investigated. This does not yield complete and accurate results as it cannot comprehensively determinate the causal relations. In our approach, the states of both structure G and parameter Θ are inferred to give a full description of current network.

As we deal with probability values, we can also use predictive mechanisms to determine the effect of one parameter on the entire network. The generation is done according to the knowledge of the network behavior. The hypothesis that is chosen is then sent to the Bayes output generator. This generator then computes the best possible data rates, sampling, re-scheduling information for optimal performance and stability. The cause and effect of parameters chosen for the next time slice is determined and the control commands to the actuators are preset accordingly. The commands will eventually be updated according to network's performance and behavior. This entire process is possible because of the information acquisition and fusion. Our method also incorporates the possibility of an non-Gaussian input to the BN. We employ the Gaussian Mixture Model, which is a probability density function represented as the weighted sum of the Gaussian component densities. It is given by

$$p(x|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \sigma_i) \tag{3}$$

where x is the D-dimensional data vector of measurements or features, w_i is the mixture weights and $g(x|\mu_i, \sigma_i)$ $\forall i = 1, 2, ...M$ are the sub-densities (components of the non-Gaussian input), μ_i and σ_i are the mean and covariance respectively. By splitting the incoming probability density functions in this manner of sub-components, our information fusion scheme can tackle non-Gaussian distributed data.

In order to further help in accurately approximating the true distribution, we use a

feature selection algorithm which selects the most relevant features to be used in determining the posterior distribution. In the feature selection and update algorithm, let F be a pool of features to be selected from. The features are specific to the CPS application and can be obtained from the traffic information, system behavior, observed noise etc. At the start of the selection process, the pool is empty as no features are selected for analysis. Later, upon adding new features f_i and removing low impact features f_k i.e. features whose extraction and use in further information polling process will not yield a better cost benefit for the strategy selection, we obtain a new feature pool F_{new}. Γ is the causation co-efficient that determines the features to be selected and input to the next level in generating causal graphs. The causation co-efficient determines the strength or the impact among a set of features. It is empirically determined under the constraint of maximizing the utility function and has an application specific lower bound $l.bound$. Feature selection is described in Algorithm 6.

Algorithm 6: Feature Selection and Update Process

Input: Incoming traffic stream, Observed system state, system noise
Output: System and network features, Task prioritization, profiling task dependency
Initially $F = \Phi$
while $i < F$ **do**
  $\forall f_i \notin F$
  Perform $f_i \bigcup F$
  Update $F \rightarrow F_{new} = F \cup f_i$
  Update $F_{new} - f_k \; where, \; f_k \in \Gamma < l.bound$
  $*remove\ low\ impact\ features$
end
return $F = F_{new}$
determine $task\ T\ \forall\ f_i \in F$

Feature selection has a greedy approach and sometimes might not give the desired output. In other words, the feature selection algorithm may reach an equilibrium thereby,

being unable to detect undetectable and unidentifiable attacks. In such cases, a simple solution is to multiply the causation among the variables with random values in order to make the set of values more uniformly distributed. By doing so, we make sure that new features can also get added into existing features (parameters) in the feature pool. This approach adds pruning to the fusion architecture and makes it more accommodating and fair; thus, making it more accurate. It must be noted that the feature selection algorithm chooses parameters that are network specific and not cloud specific and they are still very much applicable to large scale cloud deployments such as is the case with Amazon cloud services.

Time complexity analysis: Calculation of symmetric uncertainty can be done in linear time depending on the number of features k used from the traffic trace. Since the feature selection algorithm is iterative and at each instant at an average half of the features in the pool are eliminated. This gives us a time complexity of $O(NlogN)$. Hence, the overall time complexity is $O(kNlogN)$.

V. WORKFLOW MODELING - QOS BASED ROBUST TASK SCHEDULING

Task scheduling based on QoS requirements is an important factor that goes into determining the overall utility metric of the player. In this paper, we describe the important metrics from an defenders perspective in providing secure and reliable QoS.

Upon profiling, prioritizing, and determining the tradeoffs of performing some tasks over others, we need to determine an appropriate makespan (scheduling length) to reduce the overall latency. Since decisions about the order of task completion are made based on the QoS requirements of both the user and CPS application, we generically use three basic QoS requirements cost, latency and reliability upon which we determine the effectiveness of the performance. The overall latency is affected by the delays in task completion, cost is determined by the number of service instances (Virtual Machines) on the cloud used for information processing, the migration of information among Virtual Machines (VMs) and other costs specified in Service Level Agreements (SLAs). Finally, reliability is the overall availability of the CC platform to serve the QoS requirements of the user or the CPS application. Of the three QoS requirements, cost is the most volatile and sensitive factor that is easily affected by the other two QoS requirements. Hence, cost as a determinant of effective workflow modeling under delays is discussed in detail under latency and reliability.

The function of the workflow scheduler is to allocate all tasks in the workflow to available resources (VMs) to generate a concrete workflow that minimizes latency and cost and maximizes the reliability. The scheduling model also incorporates user-defined thresholds for QoS parameters, i.e., task completion deadlines, budget for application processing, and the lower bound on reliability expected from the CC platform. Hence, the objective of the scheduler is to find a schedule that optimizes the user-preferred QoS parameters and satisfies all the user-placed QoS restrictions.

Because of the presence of precedence relations among the tasks, we model the work-

flow as a Directed Acyclic Graph (DAG) $G = (V, E)$. Let n be the number of tasks in the workflow. The set of nodes $V = \{T_1, T_2, ..., T_n\}$ corresponds to the tasks of the workflow. The set of edges E represents precedence relations between tasks. An arc is in the form of (T_i, T_j) where T_i is called the parent task of T_j and T_j is the child task of T_i with each task having an execution time e_{t_i}. A child task can only be executed until all of its parent tasks have been completed. The set of parent tasks of T_i is denoted by $par(T_i)$, and the set of child tasks by $chi(T_i)$. Each task T_i, where $1 \leq i \leq n$, has a computational domain $S_i = \{s_i{}^1, s_i{}^2, ..., s_i{}^{m_i}\}$ where, $s_i{}^j$ represents a service instance (VM) provided by the cloud service provider and m_i is the total number of available service instances for T_i. The properties of a service instance can be denoted as a tuple $(s_i{}^j.r, s_i{}^j.l, s_i{}^j.c)$ where the terms in the tuple correspond to the reliability, latency and cost of the chosen service instance. Task precedence specifies the execution order of tasks that a feasible solution must/can satisfy. We now define the three QoS constraints based on which the DAG is constructed.

Reliability constraints: The reliability of the generated concrete workflow must not be smaller than a user-defined variable $reliability\ constraint$. In other words, given a schedule $Sc \in (Sc_1, ..., Sc_n)$ which implies T_i is executed by $s_i{}^{Sc_i}$, if $Sc.reliability$ is the reliability of a schedule Sc then Sc satisfies the user defined $Reliability$ only if

$$min\ s_i{}^{Sc_i}.\ r \geq\ Reliability \tag{4}$$

Latency constraints: Total execution time of the workflow must not be larger than the application specific $Deadline$.

$$Sc.Latency = max\ s_i{}^{Sc_i}.\ l \leq\ Deadline \tag{5}$$

Cost constraints: Given a schedule $Sc \in (Sc_1, ..., Sc_n)$, the total cost of $Sc.cost$ must not be larger than the variable cost as given by the user.

$$Sc.Cost = \sum s_i^{Sc_i}.\ c \leq\ Cost \tag{6}$$

We solved the problem of scheduling by a heuristics aided schedule selection method. Since there are multiple QoS parameters that affect and determine the stability of a CPS, heuristically providing insights into their requirements will yield different solutions at different time instants. Employing this solution instead of a singly selected scheduling mechanism at the start of the process will naturally yield accurate estimation results and help us in providing security, reliability and optimized cost. Since this scheduling method is aided by the decision output from our SNFG, employing a heuristic method is almost equivalent to an exhaustive approach as decisions taken by players are based on incomplete information which requires considering multiple input parameters. The following are a few of the heuristics.

Reliability heuristic: This heuristic promotes the scheduler to select a scheduling policy that ensures reliability. The reliability requirements are determined by the most critical system in the hybrid system or by the system that causes communication bottlenecks by using most of the bandwidth. A drop in the communication packets can render the system unstable, thereby, heavily affecting the reliability of the system. Hence, to determine the level of importance to be given to a scheduling policy, we have to determine the reliability criticality $(RC_{i,j})$ by (7).

Time heuristic: The Time heuristic biases the selection of the service instances with shorter execution time. Denoted by $(TC_{i,j})$, it can be determined by (8).

Cost heuristic: The Cost heuristic biases the selection of the service instances with lowest overall cost. Denoted by $(CoC_{i,j})$, it can be determined by (9).

$$RC_{i,j} = \frac{s_i^{\ j}.r - min.rel_i\ +\ 1}{max.rel_i - min.rel_i + 1} \tag{7}$$

$$TC_{i,j} = \frac{max.time_i - {s_i}^j.l + 1}{max.time_i - min.time_i + 1} \tag{8}$$

$$CoC_{i,j} = \frac{max.cost_i - {s_i}^j.l + 1}{max.cost_i - min.cost_i + 1} \tag{9}$$

where *min.rel* is the lowest reliability value of the service instant for a particular node (schedule) and *max.rel* is the maximum of the values in (7); *min.time* is the lowest latency value of the service instant for a particular node (schedule) and *max.time* is the maximum of the values in (8); *min.cost* is the lowest cost incurred in using the service instant for a particular node (schedule) and *max.cost* is the maximum of the values in (9).

A. TASK SCHEDULING ALGORITHM

We would now like to describe our task scheduling algorithm that takes into account the above mentioned QoS constraints. Task scheduling on the VMs is extremely critical as it determines the overall makespan. It is also instrumental in meeting with the application specific delay constraints. It must be noted that Scheduling and Load Balancing on the available resources are two distinct concepts of heterogeneous computing. Our Secure Task Scheduling Algorithm (STSA) starts by prioritizing the tasks based on the importance of their processing on the overall system behavior. This is done by determining the risk associated with the on-time task completion in addition to an attack detection by the SNFG. The attack risk probability γ_a is modeled as a negative exponential given by

$$\gamma_a = 1 - e^{\lambda(U_A - U_D)} \tag{10}$$

where, U_D and U_A are the defense and attack strategies of the defender and attacker. Equation (10) implies that the risk probability grows as the difference between the two utilities grows. Then, using the concepts of queuing theory, the scheduler calculates the mean

service time on a VM on the cloud, the average waiting time for a schedule to be cleared off the queue (waiting time on the queue W_q) and schedules the incoming tasks accordingly. The tasks are prioritized based on γ_a and its out/in degree (dependency on other tasks). This ensures that resources are provided to the tasks with high urgency. However, there is a down side to this approach. There could be tasks that might never meet the emergency criteria and remain in the queue forever. In order to break such a deadlock, we define a preset maximum allowable time threshold ψ beyond which it has to be allotted a computing resource. ψ is determined using τ_{mad} which is the maximally allowable delay before which QoS requirements for the CPS have to be serviced. This τ_{mad} is the $Deadline$ mentioned previously in equation (5).This threshold value is obtained by taking as reference, the largest allowable waiting time on a task based on previous experiences. Algorithm 7 is our Secure Task Scheduling Algorithm (STSA) under the QoS constraints.

B. ALGORITHMIC COMPLEXITY

We now determine the time complexity of our scheduling algorithm which is based on a DAG. The complexity is computed based on the number of tasks t, number of edges e, and number of nodes n.

1. There are t tasks in the DAG for parallel computing. For each task t_i, the computation of task priority can be done in $O(n + deg)$, where deg is the maximum in degree or out degree of the task. Therefore, the overall time taken to compute the task priority is $O(t(n + deg))$.

2. After computing the task priority, a efficient sorting algorithm can be used to sort the tasks in logarithmic time $O(tlogt)$.

3. To determine the task completion time for all the tasks for all the nodes based on the QoS requirements, the complexity is $O(n * deg)$.

Algorithm 7: QoS satisfying Secure Task Scheduling Algorithm (STSA)

```
Input: Attack information from SNFG, incoming set of tasks (T)
Output: Task scheduling, Task prioritization, DAG
for ∀{t_i} ∈ T do
    *TASK PRIORITY TO GENERATE SCHEDULE {Sc} *
    generate DAG {G}
    while G do
        compute attack risk probability γ_a ∀ {T}
        compute edge weights w for DAG {G}
        w = task_priority{t_i, t_j} + γ_a(T)
        {G'} = sorted {G}
    end
    update {G} = {G'}
    ***SCHEDULING {Sc} ON VMs ***
    determine available space on VMs, W_q
    calculate makespan, speedup, memory requirement
      of task on VM, τ_mad
    allocate VMs to tasks t_i ∈ {T} ; Schedule{Sc}
    determine completed tasks t_k
    update Schedule{Sc'} = {Sc} − t_k
    return Task completion time, Sc
    END
end
```

4. To allocate the tasks to all the resources, in the worst case scenario, the execution time is $O(n)$. In order to determine the overall scheduling time we have to consider the completion time and the allocation time together. Hence, the time complexity is $O(t(n*deg+n))$.

5. The overall time complexity of the entire algorithm is $O(t(n*deg+n)+tlogt+t(n+deg))$.

VI. RESULTS AND ANALYSIS

To evaluate our algorithm with other existing algorithms, we conducted the experiments on a wireless networked control system (WNCS) plant. We made use of both PiccSim [4] and NS2 to simulate WNCS. PiccSim is a simulation software that models the control plant. This model is the input to the simulator. The sensor network of 100 nodes is simulated using NS2. PiccSim is MATLAB based with a feature that can link the control model in MATLAB to the network model in NS2. A simulator was used to generate attack traffic owing to the unavailability of real CPS data under security attacks. We used Amazon's EC2 instances to run our experiments. We used 4 small instances 3 medium instances and 2 large instance with RAM sizes 4Gbytes, 8Gbytes and 16Gbytes and with 4 cores, 8 cores and 16 cores. The number of tasks that were generated for the performance analysis was 800, 1600 and 2400, 3200 and 4800 for each run of experiments with variable dependencies of 40%, 60% and 80% of the tasks as dependent tasks. The tasks corresponded to secure aggregation of the data, state estimation, attack detection, resource allocation, scheduling, decision dissemination etc. We compared our algorithm's performance with that of two very well known and highly used scheduling algorithms in heterogeneous systems - Heterogeneous Earliest Finish Time algorithm HEFT [3] and Dynamic-Level Scheduling DLS [21] algorithms and one Security Driven List Scheduling (SDS) [22] algorithm. In addition to these algorithms, we also compare the speedup of our approach with two other works, Critical Path Genetic Algorithm (CPGA) [14] and the Duplication Scheduling Heuristic Algorithm (DSH)[19]. Since both these works are heuristic algorithms, we wanted to compare our work to see the overall speedup. We used a modified version of the HEFT and DLS algorithms to allow them to deal with security. Although these algorithms are intended to schedule tasks with security requirements, they make no effort to optimize the quality of security.

The performance metrics chosen for the comparison are the makespan, resource utilization, speedup - computed by dividing the sequential execution time which is the sum of the computational costs and security requirements of the tasks by the parallel execution time (makespan), total costs and overall reliability. Sequential execution time is computed on a uni-processor where the speed up does not show improvement beyond a point. The parallel execution time is computed by assigning multiple computational resources to the dependent tasks where speedup is determined by algorithm's effectiveness. The entire comparison will be done in order to obtain a qualitative understanding of the performance and drawbacks of the proposed scheduling algorithm.

We first start the discussion of our results with the performance of our modified SNFG that has its gluing functions based on the state transitions as observed using a time-varying DBN.

A. SEMI NETWORK-FORM GAME

Figure 1 shows the incentive of an attacker (utility function) versus the number of nodes of the network he/she has captured. This analysis is useful in determining the strength of the algorithm when the attacker's access to the system is considerably large. The game played by the defender now is dependent on the QoS requirements of the WNCS and the system operator (defender). As shown in the figure, p is the probability of the presence of the attacker in the system. The probability is varied to give weight to the consideration of an attackers presence vs system faults. It is then compared to the SNFG game played when the presence of an attacker has a probability value of 0.85. This was done in order to test the effectiveness of our algorithm in determining the presence of an attacker and choosing appropriate strategies. As we can see from the graph, in comparison with the case where $p = 0.85$ where there is no game theoretic attack solution being applied because of lack of information about system state, an attacker goes undetected until he captures 50% of the nodes. As opposed to that, our scheme works exceedingly well even when half of

the networks nodes are compromised. Even when 50% of nodes are captured, the incentive of the attack is kept below 0.2, which reflects that the cost involved to perform this attack is very high. We assume a reasonable attacker would not conduct an attack causing losses to himself.

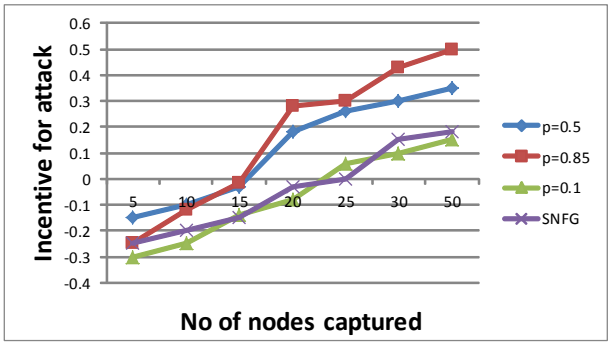

Figure 1. Mitigating attack by reducing attack incentive

B. SPEEDUP

We considered speedup as another performance metric to study the effectiveness of our scheduler in task prioritization and resource allocation. Speedup is estimated as $S = T(1)/T(C)$. i.e. the time taken to execute a set of tasks on a uni-processing unit vs the time taken to perform same set of tasks using C processing units (VMs). We compared our results with Critical Path Genetic Algorithm (CPGA) [14] and the Duplication Scheduling Heuristic Algorithm (DSH). As we can see from the figure 2, our algorithm outperformed the other two algorithms except in the case of 4VMs for CPGA. This is because of the availability of extra processing which aids the genetic algorithm in [14].

However, this makes [14] static as any reduction in the VMs will affect their performance. Hence, our algorithm scales smoothly over variable number of VMs. Our algorithm

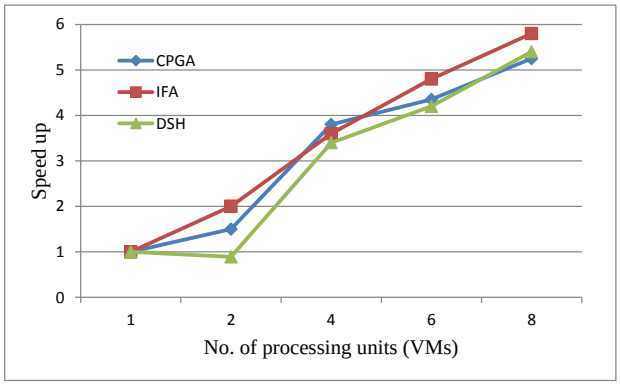

Figure 2. Speed up comparison with heuristic scheduling algorithms

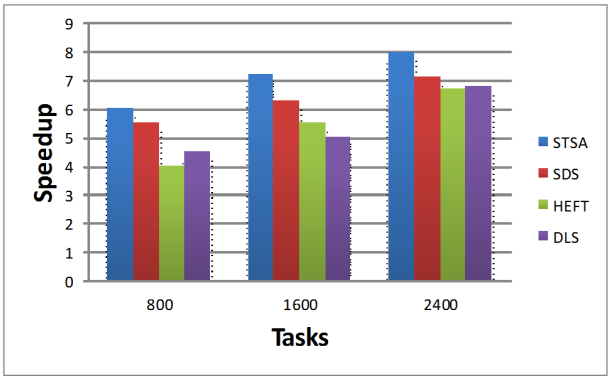

Figure 3. Speedup with security driven scheduling algorithms

showed a linear increase in speedup despite bottlenecks such as bandwidth constraints, whereas the other two algorithms performed sub-linearly. Hence, our scheduler outputs decisions upon considering the overall performance increase, instead of just considering latency reduction. This shows that our scheduling heuristic algorithm is robust to attack induced constraints as it is aided by our SNFG. We will now discuss the performance of our approach in comparison to the approaches that have security considerations. As we can see from figure 3, our approach outperforms the other approaches for all different sets of tasks.

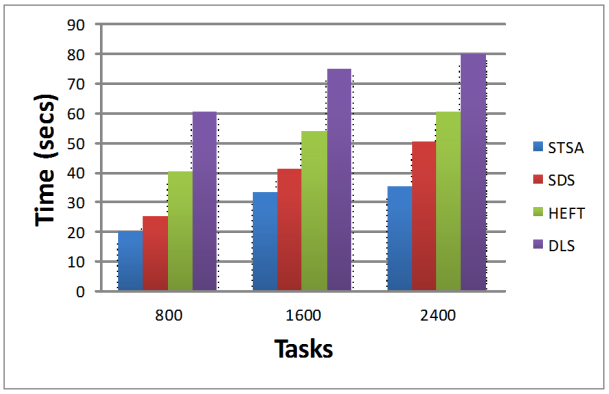

Figure 4. Makespan of the algorithms for different sets of tasks

As speedup determines the effectiveness of an algorithm's ability to make decisions about resource allocation under both computational and task priority contention, higher speedup implies better performance capabilities.

C. MAKESPAN

Experiments about makespan were conducted in comparison to the scheduling algorithms that incorporated security. Each task arriving on the cloud has security requirements as per its priority which is based on the impact its completion has on the overall system reliability. In order to determine such an impact, based on our SNFG, we varied the probability of the presence of an attacker to find out the overall system risk.

Figure 4 shows the simulation results of our approach in comparison with the three algorithms SDS, modified HEFT, and modified DLS on the cloud. We can observe from figure 4 that our algorithm significantly outperforms the rest in terms of the overall makespan. We see an improvement in the performance of our algorithm because of the added optimization in parametric selection before the scheduling algorithm is used. Our SNFG chooses strategies which help the scheduler to understand the system requirements and behavior better. The total makespan is given in seconds and shows the total time taken for the

scheduler to allocate the tasks to the VMs for their processing (completion).

This does not mean that decisions will be taken only after all the tasks have been finished. Based on the DAG, dependent tasks will be finished (based on their security priority) and corresponding control decisions will be output and sent back to the actuators. We performed the experiments for tasks ranging from 800 to 2400 with a maximum of 80% of dependent tasks. However, the makespan reduces as the dependency among the tasks reduces. Also, we observed that for a task dependency of about 75 - 80%, the makespan was optimal as sequential task completion allowed for easier decision making in resource allocation. Also, the other algorithms require some amount of security overhead which improves the makespan. As our algorithm does not require such an additional security overhead, makespan can be further reduced.

We now provide an in depth analysis of the effectiveness of our algorithm in limiting the increase in makespan with respect to number of nodes captured by the attacker, the probability of the presence of an attacker p, and the false positive percentage. As we can see from figure 5, varying the values of p from 0.1 to 0.85 which indicate the confidence in a defender's inference about the presence of an attacker and consequently the strength of the attack, results in a gradual increase in the total. The reason we do not choose values of $p < 0.1$ is because; using the sampling a considerable number of states for lower p values go unnoticed or are not frequented, thereby, failing in the detection of the attacker. This unfortunately is a drawback of choosing any sampling algorithm. To test the strength of the SNFG implementation, we chose the value of p a 0.6. This was done in order to make sure that the defender strategies are not biased completely on the presence of an attacker. We intended to include a certain degree of uncertainty to better mimic the lack of complete system knowledge for the defender. In addition to that, false positives will be present and an intelligent and tactful attacker will utilize such discrepancies arising from system noise, packet drops, network delays etc. as we can see from the figure 5, our scheme is effective in determining the defender move spaces even though the attacker is assumed to be in level

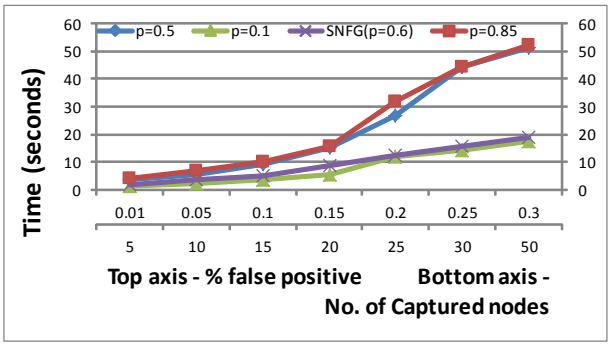

Figure 5. Increase in makespan vs p vs % false positives

$k-1$. This goes on to show the effectiveness of our overall STSA in reducing the makespan which in turn reduces the overall latency.

A. Makespan based on task priority.

As another metric to determine the efficiency of our algorithm, we study the effect of task dependency while prioritizing the tasks based on their criticality. From figures 6 and 7, for different task sets with variable dependencies, we see that the use of our STSA out performs the cases of regular scheduling such as round robin or earliest deadline first as implementable in Amazon cloud. A reduction in makespan is a result of the manner in which our algorithm alters the schedule length and the scheduling policy based on the defender move spaces, which in turn considers the attacker's perceived move space at level $k-1$. A reduction in the overall makespan corresponds to the reduction in the processing latency. From figures 6 and 7 and 8, we can infer that our algorithm services the high priority tasks on the more powerful VMs by allocating more number of cores for the task completion. This results in higher resource utilization while keeping the utility function optimal at each stage of the game over multiple such stages. This results in an overall reduction in the cost required to maintain the QoS requirements of the CPS and consequently its stability and reliability even under security attacks.

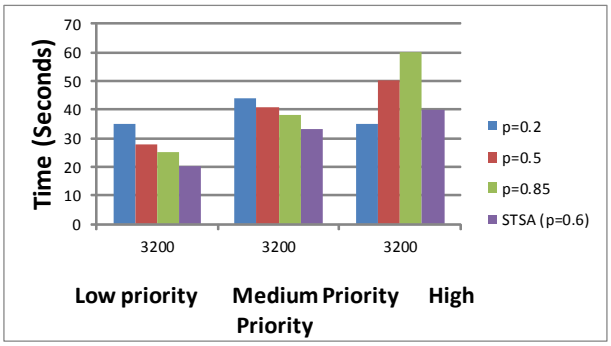

Figure 6. Increase in makespan based on task priority

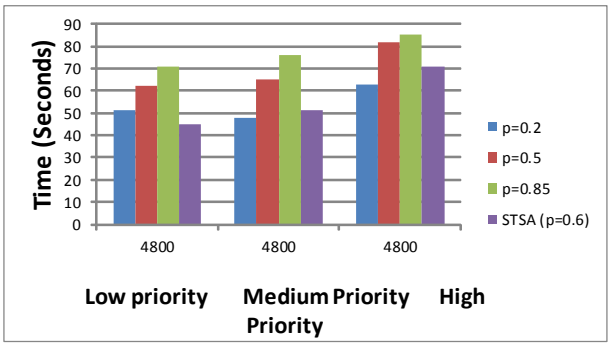

Figure 7. Increase in makespan based on task priority

D. RESOURCE UTILIZATION

In order to determine the scheduling efficiency of our proposed algorithm in comparison with the other algorithms, we determined the overall utilization of the resources (VMs) that were employed. This study is important as when using a cloud platform, the resources become pay-per-use. Hence, assumptions about abundant computational resources cannot be assumed. Also, it is imperative from the users' perspective to utilize the avail-

able resources that are already being paid for. Figure 8 shows the comparison of resource utilization by our algorithm with the other algorithms for 800, 1600 and 2400 tasks. We see an improvement in the performance of our SNFG aided scheduling algorithm because it takes as inputs the decisions from a game theoretic approach which looks to maximize the utility factor of the defender. Maximizing the utility factor is conditioned upon the defender's observability of the cyber component as well, i.e. the defender is able to thwart any DoS attack on the VMs. This gives more flexibility to choose the VMs for resource allocation. It also eliminates the need for inter VM migration of the information upon the detection of an attack.

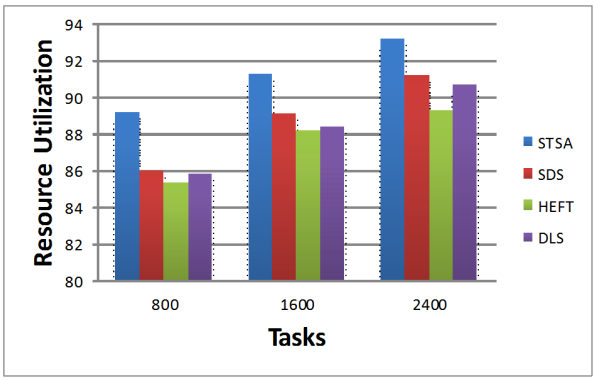

Figure 8. Utilization on VMs

A. Ductility.

We now study the effectiveness of our algorithm based on the ductility as described in [10]. We service the tasks in our scheduling algorithm based on their priorities, which is reflective of their criticality. Critical tasks correspond to the ones where QoS determining computational latency must be low. Other than such tasks, information pooling from sensors upon the detection of an attacker and the inference of his attack space which includes

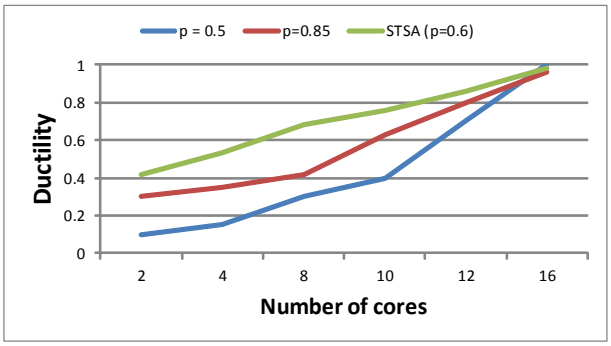

Figure 9. Ductility of the scheduling algorithm

both control theoretic state estimate disturbances and denial of service attacks; fall under high criticality/ priority tasks. As discussed before, we service the low, medium and high priority tasks. Their criticalities can be assignment as $< 0, 1, 2 >$ to ensure a sense of order for queuing in the server queue. We studied the effects of task disorder to test the efficiency of our algorithm in reverting back to satisfying the QoS requirements of the CPS application. Figure 9 shows that our STSA algorithm achieves significantly better performance compared to the cases of $p = 0.5, 0.85$ for the HEFT and DLS algorithms, even when a small number of cores are made available (about 1.5 and 4.5 times better performance). With an increase in the number of cores, we can see that the performance difference between the three algorithms in comparison decreases. This is because when the number of cores reaches a large enough value sufficient to schedule the different task sets, all the tasks are ably and appropriately serviced by the virtual machines. This is largely due to the approximate nature of our heuristics algorithm. for different task sets with different dependencies among them, HEFT and DLS can perform worser than our algorithm owing to the fact that they do not consider the complications that arise in dependent tasks.

E. RELIABILITY

Reliability is calculated based on the total number of times a service request is fulfilled to the total number of times the service is issued over the entire run time of the algorithm. This is used to determine how effective our scheduling algorithm is in resource allocation based on task prioritization which affects the task execution time. As reliability metric can be calculated in many ways, in our paper, we calculate it based on the number of times correct state estimate was sent back to the actuator, and the number of times an attack was countered correctly after detection under delay constraint τ_{mad}. As we can see from table 1, our STSA out performs the other three algorithms in terms of reliability guaranteed. We attribute this to the fact that our scheduling algorithm is backed by an effective game theoretic approach which works even under an attack. Although SDS is a security driven approach, it considers the use of trust and security approaches such as authentication confidentiality and integrity. Their scheduler is based on the determination of the trust level of only the service provider. However, in a real time application like CPS, an attack could take place in both the physical and the cyber domain. The column corresponding to $> \tau_{mad}$ in table 1 shows the ineffectiveness of other schedulers to provide robustness when an attack cannot be detected or cannot be mitigated. As compared to them, our scheme performs admirably well even when the delay is large, thus providing robust QoS constrained scheduling for CPS.

Table 1. Reliability Percentage

Algorithm	$\leq \tau_{mad}$	$> \tau_{mad}$
STSA	99.6	98.7
SDS	98.3	86.3
HEFT	93.4	84.2
DLS	93.2	85.1

VII. CONCLUSIONS

In this paper, we have provided an extension to the semi-network form game using a time-varying DBN (TVDBN) which allows us to play multi-player games against an attacker with minimum or zero knowledge about the system state. Computational load of the TVDBN was solved by using our proposed feature selection mechanism to give on-line conditional probabilities which are otherwise very difficult to obtain. We conducted our experiments to validate the effectiveness of our approach and found out that even under constraints our approach performed linearly or better where other methods had a sub-linear performance indicating the robustness of our algorithm. In future, we will conduct more extensive experiments on a larger data set while taking more complex and detailed QoS requirements into consideration. We would also test the efficiency of our algorithm on multiple CPS applications. Based on results obtained, and the scalable nature of our algorithm, we are confident in reproducing similar results under extensive and scaled out testing.

REFERENCES

[1] S. Abrishami, M. Naghibzadeh, and D. H. Epema. Cost-driven scheduling of grid workflows using partial critical paths. *Parallel and Distributed Systems, IEEE Transactions on*, 23(8):1400–1414, 2012.

[2] H. Arabnejad and J. Barbosa. List scheduling algorithm for heterogeneous systems by an optimistic cost table. 2013.

[3] C. Augonnet, S. Thibault, R. Namyst, and P.-A. Wacrenier. Starpu: a unified platform for task scheduling on heterogeneous multicore architectures. *Concurrency and Computation: Practice and Experience*, 23(2):187–198, 2011.

[4] M. Bjï£¡rkbom, S. Nethi, and T. Kohtamï£¡ki. Biometrics: Promising frontiers for emerging identification market. Technical Report PiccSIM version 1.12, Aalto University, School of Science and Technology, 2011.

[5] T. D. Braun, H. J. Siegel, A. A. Maciejewski, and Y. Hong. Static resource allocation for heterogeneous computing environments with tasks having dependencies, priorities, deadlines, and multiple versions. *Journal of Parallel and Distributed Computing*, 68(11):1504–1516, 2008.

[6] S. K. Garg, C. S. Yeo, A. Anandasivam, and R. Buyya. Environment-conscious scheduling of hpc applications on distributed cloud-oriented data centers. *J. Parallel Distrib. Comput.*, 71(6), June 2011.

[7] K. Z. Gkoutioudi and H. D. Karatza. Multi-criteria job scheduling in grid using an accelerated genetic algorithm. *Journal of Grid Computing*, 10(2):311–323, 2012.

[8] X. He, X. Sun, and G. von Laszewski. Qos guided min-min heuristic for grid task scheduling. *J. Computer Science Technology*, 18(4), July 2003.

[9] J.-K. Kim, S. Shivle, H. J. Siegel, A. A. Maciejewski, T. D. Braun, M. Schneider, S. Tideman, R. Chitta, R. B. Dilmaghani, R. Joshi, A. Kaul, A. Sharma, S. Sripada, P. Vangari, and S. S. Yellampalli. Dynamically mapping tasks with priorities and multiple deadlines in a heterogeneous environment. *J. Parallel Distrib. Comput.*, 67(2), Feb. 2007.

[10] K. Lakshmanan, D. de Niz, R. Rajkumar, and G. Moreno. Resource allocation in distributed mixed-criticality cyber-physical systems. In *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, pages 169–178, June 2010.

[11] R. Lee and D. H. Wolpert. Game theoretic modeling of pilot behavior during mid-air encounters. *CoRR*, abs/1103.5169, 2011.

[12] A. Mandal, K. Kennedy, C. Koelbel, G. Marin, J. Mellor-Crummey, B. Liu, and L. Johnsson. Scheduling strategies for mapping application workflows onto the grid. In *High Performance Distributed Computing. HPDC-14. Proceedings. IEEE International Symposium on*, 2005.

[13] S. H. Nielsen and T. D. Nielsen. Adapting bayes network structures to non-stationary domains. *International Journal of Approx. Reasoning*, 49(2):379–397, Oct. 2008.

[14] F. A. Omara and M. M. Arafa. Genetic algorithms for task scheduling problem. *Journal of Parallel and Distributed Computing*, 70(1):13 – 22, 2010.

[15] F. Pasqualetti, F. Dorfler, and F. Bullo. Attack detection and identification in cyber-physical systems. *Automatic Control, IEEE Transactions on*, 58(11):2715–2729, Nov 2013.

[16] M. Rahman, R. Hassan, R. Ranjan, and R. Buyya. Adaptive workflow scheduling for dynamic grid and cloud computing environment. *Concurrency and Computation: Practice and Experience*, 25(13):1816–1842, 2013.

[17] L. Song, M. Kolar, and E. P. Xing. Time-varying dynamic bayesian networks. In *NIPS*, pages 1732–1740, 2009.

[18] S. Song, K. Hwang, and Y.-K. Kwok. Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling. *Computers, IEEE Transactions on*, 55(6):703–719, 2006.

[19] H. Topcuouglu, S. Hariri, and M.-y. Wu. Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Trans. Parallel Distrib. Syst.*, 13(3), Mar. 2002.

[20] A. Tucker and X. Liu. A bayesian network approach to explaining time series with changing structure. *Intelligent Data Analysis*, 8(5):469–480, Oct. 2004.

[21] W. Wang, G. Zeng, D. Tang, and J. Yao. Cloud-dls: Dynamic trusted scheduling for cloud computing. *Expert Systems with Applications*, 39(3):2321–2329, 2012.

[22] T. Xiaoyong, K. Li, Z. Zeng, and B. Veeravalli. A novel security-driven scheduling algorithm for precedence-constrained tasks in heterogeneous distributed systems. *Computers, IEEE Transactions on*, 60(7):1017–1029, 2011.

[23] X. Xuan and K. Murphy. Modeling changing dependency structure in multivariate time series. In *Proceedings of the 24th international conference on Machine learning*, pages 1055–1062. ACM, 2007.

[24] J. Yu, R. Buyya, and C. K. Tham. Cost-based scheduling of scientific workflow ap-

plications on utility grids. In *e-Science and Grid Computing, 2005. First International Conference on*, 2005.

[25] S. Zheng, W. Shu, and L. Gao. Task scheduling using parallel genetic simulated annealing algorithm. In *Service Operations and Logistics, and Informatics. IEEE International Conference on*, 2006.

[26] X. Zhu, X. Qin, and M. Qiu. Qos-aware fault-tolerant scheduling for real-time tasks on heterogeneous clusters. *Computers, IEEE Transactions on*, 60(6):800–812, 2011.

SECTION

4. CONCLUSION

This dissertation presents information fusion algorithms for Wireless Sensor Networks and Cyber Physical Systems. First, an algorithm for data aggregation in an underground mine application was proposed. Next, a multi-layered information fusion architecture to detect security attacks in a wireless networked control system was presented. Thereafter, an information fusion architecture to schedule variable loads of information from a cyber physical system was proposed. Finally, the dissertation was concluded by presenting an extension to a game theoretic approach which was coupled with a robust scheduling mechanism to guarantee QoS in cyber physical systems.

The fusion architecture proposed in Paper I estimates the unknown noise parameters in the sensor network. An efficient method for calculating these values on power constrained sensor motes was developed as the computations do not consider any matrices and inverses. The efficiency of the scheme lies in the multiple levels of aggregation performed on the sensor motes. The novelty of the architecture lies in its ability to predict accurate state estimates even under varying noise parameters. An n-step Kalman filter was presented along with the methodology to model changes in system and observation noise.

The proposed approach in Paper II introduced an information acquisition and fusion scheme for Wireless Networked Control System (WNCS), where, both feature extraction and decision making are used in securing and stabilizing the system. Theoretical bounds of time delays were provided in order to ascertain system stability. It was shown that we can mitigate the effects of security attacks and still provide the WNCS with operational stability. A novel time-varying Dynamic Bayesian Network was proposed to show how changes in the network behavior can be efficiently observed and decisions made. Based on

these decisions, the control messages had been altered to suit the requirements of WNCS.

In Paper III a causality based information fusion architecture to bridge the gap that exists between scalable processing with minimal latency for time-critical information dissemination in a CPS was proposed. The effects of self-similarity and long range dependence in predicting the traffic bursts over time intervals was thoroughly studied. An information fusion architecture that effectively manages task scheduling using load balancing on the cloud was proposed.

In Paper IV an extension to the semi-network form game using a time-varying DBN (TVDBN) which allows us to play multi-player games against an attacker with minimum or zero knowledge about the system state was presented. Computational load of the TVDBN was solved by using the proposed feature selection mechanism to give on-line conditional probabilities which are otherwise very difficult to obtain. An in-depth study of the manner of human interactions in the semi-network for game was presented. Extensive simulation results were presented to cover as many test cases as possible to eliminate the drawbacks of the heuristic nature of the scheduling algorithm.

BIBLIOGRAPHY

[1] S. Amin, A. Cardenas, and S. Sastry, "Safe and secure networked control systems under denial-of-service attacks," *Hybrid Systems Computation and Control*, Apr. 2009, vol. 5469, pp. 31–45,

[2] Y. Liu, M. K. Reiter, and P. Ning, "False data injection attacks against state estimation in electric power grids." ACM Conference on Computer and Communications Security, Nov. 2009, pp. 21–32.

[3] Y. Mo and B. Sinopoli, "Secure control against replay attacks," in *Communication, Control, and Computing, Allerton. 47th Annual Allerton Conference on*, 2009, pp. 911–918.

[4] A. Teixeira, S. Amin, H. Sandberg, K. Johansson, and S. Sastry, "Cyber security analysis of state estimators in electric power systems," in *Decision and Control (CDC), 49th IEEE Conference on*, 2010, pp. 5991–5998.

[5] S. Amin, X. Litrico, S. S. Sastry, and A. M. Bayen, "Stealthy deception attacks on water SCADA systems," in *Proceedings of the 13th ACM international conference on Hybrid systems: computation and control*, HSCC, 2010, pp. 161–170.

[6] R. Smith, *"A decoupled feedback structure for covertly appropriating network control systems"*. in IFAC World Congress, Milan, Italy, Aug. 2011.

[7] M. Zhu and S. Martinez, "Stackelberg-game analysis of correlated attacks in cyber-physical systems," in *American Control Conference (ACC)*, 2011, pp. 4063–4068.

[8] H. Fawzi, P. Tabuada, and S. Diggavi, "Secure state-estimation for dynamical systems under active adversaries," in *Communication, Control, and Computing (Allerton), 49th Annual Allerton Conference on*, 2011, pp. 337–344.

[9] M. H. Manshaei, Q. Zhu, T. Alpcan, T. Basar, and J.-P. Hubaux. "Game theory meets network security", ACM Comput. Surv. 45, 3, Article 25, July 2013, 39 pages.

[10] J. Mirkovic and P. Reiher. "A taxonomy of DDoS attack and DDoS defense mechanisms". ACM SIGCOMM Computer Communications Review, 34(2), April 2004.

[11] T. Peng, C. Leckie, and K. Ramamohanarao. "Survey of network-based defense mechanisms countering the DoS and DDoS problems" ACM Computing Surveys, 39, April 2007.

[12] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu. "A survey of game theory as applied to network security" In Proceedings of the 2010 43rd Hawaii International Conference on System Sciences, 2010 pages 1-10.

[13] M. E. Snyder, R. Sundaram, and M. Thakur. "A game-theoretic framework for bandwidth attacks and statistical defenses" In Proceedings of the 32nd IEEE Conference on Local Computer Networks, 2007.

[14] W. Zang, P. Liu, and M. Yu. "How resilient is the Internet against DDoS attacks A game theoretic analysis of signature-based rate limiting." International Journal of Intelligent Control and Systems, December 2007 12(4):307-316.

[15] Q. Wu, S. Shiva, S. Roy, C. Ellis, and V. Datla. "On modeling and simulation of game theory-based defense mechanisms against DoS and DDoS attacks." In Proceedings of the Spring Simulation Multiconference, SpringSim, 2010 pages 159:1-ĂŞ159:8. ACM.

[16] J. Xu and W. Lee. "Sustaining availability of web services under distributed denial of service attacks". IEEE Transactions on Computers, Feb. 2003 52(2):195-208.

[17] G. Yan and S. Eidenbenz. "DDoS mitigation in non-cooperative environments". In Proceedings of the 7th international IFIP-TC6 networking conference, NETWORKING, Singapore, 2008.

[18] T. Khirwadkar, K. C. Nguyen, D. M. Nicol, and T. Basar. "Methodologies for evaluating game theoretic defense against DDoS attacks". In Proceedings of the Winter Simulation Conference, 2010.

[19] B. Bencsath, I. Vajda, and L. Buttyan. "A game based analysis of the client puzzle approach to defend against DoS attacks". In Proceedings of the 2003 International Conference on Software, Telecommunications and Computer Networks, pages 763-767, 2003

[20] M. Fallah. "A puzzle-based defense strategy against flooding attacks using game theory." IEEE Transactions on Dependable And Secure Computing, January 2010 7:5-19.

[21] S.K. Baruah and J.R. Haritsa. "Scheduling for overload in real-time systems." IEEE Transactions on Computers, 1997 46(9):1034-1039.

[22] Giorgio Buttazzo, Giuseppe Lipari, and Luca Abeni. "Elastic task model for adaptive rate control." IEEE RTSS, 1998 pages 286-295.

[23] Giorgio Buttazzo, Marco Spuri, and Fabrizio Sensini. "Value vs deadline scheduling in overload conditions." In Proceedings of the 16th, IEEE Real-Time Systems Symposium, 1995.

[24] Dionisio de Niz, Karthik Lakshmanan, and Raj Rajkumar. "On the scheduling of mixed-criticality real-time task sets." Proceedings of the 30th Real-Time Systems Symposium, 2009

[25] M.K. Gardner and Liu J.W.S. "Performance algorithms for scheduling real-time systems with overrun and overload." In Proceedings of the 11th ECRTS, 1999.

[26] David Homan. "Designing future systems for airworthiness certification." 2009

[27] Gilad Koren and Dennis Shasha. Dover; "An optimal on-line scheduling algorithm for overloaded real-time systems." In RTSS '92.

[28] Pedro Mejia-Alvarez, Rami Melhem, and Daniel Mosse. "An incremental approach to scheduling during overloads in real-time systems." In Proceedings of the 21st, IEEE RTSS, 2000

[29] Karthik Lakshmanan, Raj Rajkumar, and John Lehoczky. "Partitioned fixed-priority preemptive scheduling for multi-core processors." Proceedings of the 21st Euromicro Conference on Real-Time Systems, 2009.

[30] Mohammad Hossein Manshaei, Quanyan Zhu, Tansu Alpcan, Tamer Bacar, and Jean-Pierre Hubaux. "Game theory meets network security and privacy." ACM Comput. Surv. 45, 3, Article 25, July 2013, 39 pages

VITA

Brijesh Kashyap Chejerla was born in Rajahmundry, India in 1984. His schooling took place in three different schools, spread across two states in India. He earned his bachelor's degree in Electronics & Communication Engineering from Sathyabama University of Science and Technology at Chennai in 2006.

Brijesh came to the Missouri University of Science and Technology in 2007, where he earned his Master of Science in Computer Engineering, in December, 2008. He then earned his Doctor of Philosophy degree in Computer Science in May 2015. While there, he worked as a research assistant with Dr. Sanjay Madria, focusing on information fusion in automated architectures for Cyber Physical Systems. Brijesh also served as a teaching assistant for the File Structures and Introduction to Databases course and mentored undergraduate students on various project topics helping them build and maintain an online database system.