Building System Immunity in Real Time

Andrei Tsvetkovski
Department of Electrical and Computer Engineering
Boston University
Email: andrej@bu.edu

Abstract

We identify research directions with regard to real-time security provisions in networks of critical real-time embedded systems, motivated by the recent advances in statistical learning algorithms.

New functionality requirements for networked critical real-time embedded and systems (CRTES) are constantly being added while retaining the classical assumptions of severe energy, computation, storage and bandwidth constraints for the system components, making design with security considerations an increasing challenge. Due to the ever higher complexity, the best strategy is that system trustworthiness be addressed at design time, as even analysis or monitoring of system health would be very difficult after the deployment for systems that were not built with these issues in mind from the outset.

The large number of nodes in an embedded network and the flexible network topology, wherein system components can dynamically join or leave the network, together with the real-time constraints warrant the use of automated and adaptive methods for system health monitoring, requiring minimal human intervention during run-time.

Trustworthiness features, including system safety, security, reliability and privacy all have the uncertainty as a common denominator. It is certainly not possible to predict all fault or misuse scenarios at design time. This is especially true when addressing security, as security related threats should be assumed to be originating from intelligent entities with a malicious intent, trying to contrive novel, polymorphic methods for compromising the system. In the presence of uncertainty and non-predictable system threats, deriving anomaly detection and prevention methods solely from deterministic event classification principles is a recipe for disaster. Probabilistic models and probabilistic reasoning about security should be preferred for properly addressing these questions.

Inference algorithms based on the statistical learning theory framework [1], the support vector machine (SVM) classifiers, have a notable potential for automated operation requiring minimal human guidance. SVMs are suitable for analysis of vast amounts of data in situations where functional dependencies between the data and the outcome of the classification are not known a priori, and particularly in situations where not all parts of the analyzed data are equally informative and explicit weights of all the features are not known, or their manual specification is not feasible. SVMs are specifically tailored for inference based on a small sample size, thus simplifying their training.

At the time SVM algorithms were introduced it was obvious that they might not be entirely suitable for real-time operation. SVMs were applied successfully to the 1999 KDDCUP intrusion detection data, but this task suggested

- *a central point of analysis*: the collected data were assumed to be communicated to a processing site with virtually unlimited processing resources; dissemination of the classification results, the associated costs and further actions were not considered.

- *off-line, post-mortem analysis of intrusion events*: the goal was to carry out a classification with maximal precision, while there were no timing restrictions or real-time requirements.

Clearly, this set of assumptions is not entirely suitable for distributed embedded systems with real-time constraints. However, recent advances in the development of statistical learning algorithms convert some of the above shortcomings of the statistical classifiers into enabling possibilities and introduce new research directions with regard to security considerations in CRTES:

Automated Model Selection

Standard applications of SVM algorithms require manual selection of a suitable kernel model and manual parameter tunning. Applications of SVMs in networked CRTES pose additional challenges in this respect. Embedded systems tend to be more tightly coupled to the physical world than traditional networked systems. Changes in the physical environment as well as changes in the network topology possibly induced by individual nodes joining or leaving the network in an ad hoc manner, imply behaviors that cannot be anticipated at design time. Additionally, CRTES are expected to work unattended and with very limited direct user interaction. These added prerequisites for CRTES present the need of automatic and adaptive updates of the utilized models.

Recent work in SVM algorithms has shown a viable possibility of automating the choice of kernel models ([2]) by learning the best kernel from available data. This further automates the task of model selection and paves the way for reducing the need of human intervention which is essential to real-time operation. Development of mechanisms for achieving self-configuration and adaptivity in the context of networked CRTES is called for in order to take the advantage of these advances in statistical learning algorithms.

Incremental On-line Learning and Unlearning

The basic SVM algorithm encompasses an off-line learning phase that takes place before the algorithm can be applied to newly generated data. Updating the acquired knowledge requires retraining. This makes the basic algorithm implementation unsuitable for applications in dynamical embedded networked systems because the amounts of data generated by such systems tend to be overwhelming, and a retraining from scratch would require excessive storage space to store the past relevant data.

But recent work shows it feasible to endow the SVM algorithm with the ability to learn and unlearn incrementally, as new data arrive ([3], [4]). This opens the door to applying an upgradable SVM learning algorithms in an on-line setting. How to perform recursive learning in real-time, or whether this can be done at all, remains to be answered.

Distributed Algorithms

One of the prominent challenges arising in the context of CRTES is: How should an embedded network comprising a large number of nodes be coordinated to perform security related tasks in a distributed manner?

The ability of a learning algorithm to learn and unlearn incrementally in time, as pointed out in [3], translates to the ability to distribute the operation in space, as chunks of data that are to be used for updating the current knowledge may be stored at physically separate sites.

Distributing the operation in space addresses at the same time the question of distributed storage of audit records. On one hand, the power and size constraints of the individual network components may limit the size of locally available storage. On the other hand, systems coupled to the real world and faced with real-world security threats are expected to store large amounts of audit data. Thus, having algorithms capable of operating on distributed data and achieving a consensus on the global system health while limiting the necessary communication would be a viable solution to this problem. More research is needed in order to address these issues. A joint approach to incremental learning and communication deserves more attention in the context of networked CRTES.

Responding in Real Time

Scalability of the basic SVM implementation to large data sets is another barrier to applying this class of algorithms in real time for large data sets. In the training phase, the time complexity of the algorithm with respect to the dataset size is $O\left(n^3\right)$ and the space complexity is $O\left(n^2\right)$ [5]. In addition, large data sets are likely to generate a large number of support vectors which implies high computational cost of each test.

However, recently there are promising results in constructing approximately optimal algorithms with provable theoretical guarantees, that can drive the time complexity down to $O\left(n\right)$ with a space complexity independent of n [5]. This is an encouraging result and outlines a new promising direction to be explored in the real-time operation of classification algorithms with applications to security.

Performance Metric, Monitoring and Event Logging

So far, most of the SVM applications dealing with security issues assume binary classification. Nevertheless, allowing more than two classes can provide a finer-grained indicator for the monitored system health. This can be used, for instance, in conjunction with the "color code" for labeling individual network entities based on their social behavior within the network.

The size of the envisioned networks prohibits storing a complete trace of the past events. Whether multi-level classification can be useful in deciding which records to keep when auditing the system operation, remains to be investigated.

REFERENCES

[1] V. N. Vapnik, *Statistical Learning Theory*. Wiley, 1998.

[2] C. S. Ong, A. J. Smola, and R. C. Williamson, "Learning the kernel with hyperkernels," *J. Mach. Learn. Res.*, vol. 6, pp. 1043–1071, 2005.

[3] D. Caragea, A. Silvescu, and V. Honavar, "Distributed and incremental learning with support vector machines," Iowa State Univ., Ames, IA, Tech. Rep. ISU-CS-TR 2000-04, 2000.

[4] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," in *NIPS*, 2000, pp. 409–415.

[5] I. W. Tsang, J. T. Kwok, and P.-M. Cheung, "Core vector machines: Fast SVM training on very large data sets," *J. Mach. Learn. Res.*, vol. 6, pp. 363–392, 2005.