Problems and Mitigation Strategies for Developing and Validating Statistical Cyber Defenses

Michael Atighetchi, Raytheon BBN Technologies
Michael Jay Mayhew, Air Force Research Laboratory
Rachel Greenstadt, Drexel University
Aaron Adler, Raytheon BBN Technologies

Abstract. The development and validation of advanced cyber security technology frequently relies on data capturing normal and suspicious activities at various system layers. However, getting access to meaningful data continues to be a major hurdle for innovation in statistical cyber defense research. This paper describes the data challenges encountered during development of the machine learning approach called Behavior-Based Access Control (BBAC), together with mitigation strategies that were instrumental in allowing R&D to proceed. The paper also discusses results from applying a spiral-based agile development process focused on continuous experimental validation of the resulting prototype capabilities.

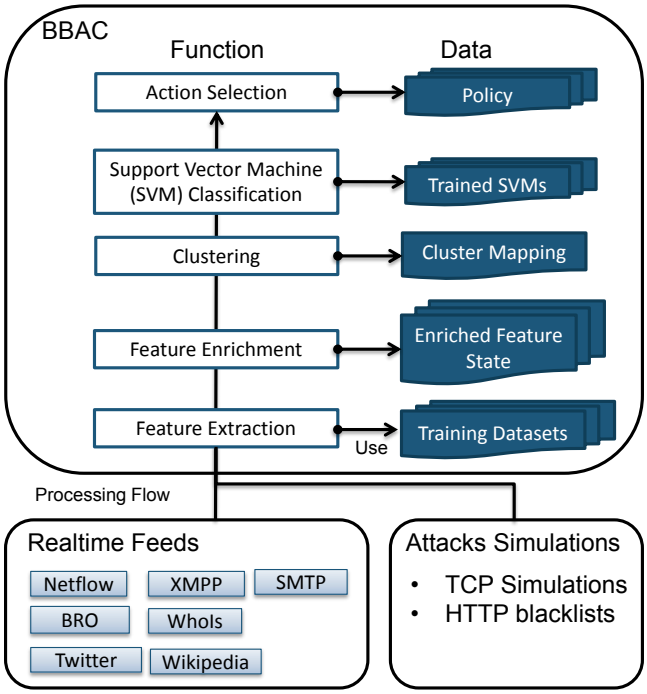

Figure 1. BBAC is a data-intensive system that turns real-time feeds into actionable information through a combination of unsupervised and supervised machine learning (clustering and SVMs).

1. Introduction

Enterprise business processes are more connected than ever before, driven by the ability to share the right information with the right partners at the right time. While this interconnectedness and situational awareness is crucial to success, it also opens the possibility for misuse of the same capabilities by sophisticated adversaries to spread attacks and exfiltrate or corrupt critical sensitive information. This is particularly true for an insider threat scenario in which adversaries have legitimate access to some resources and unauthorized access to other resources that is not directly controlled by a fine-grained policy.

BBAC augments existing authorization frameworks, such as Firewalls, HTTP proxies, and application-level Attribute Based Access Control [1] to provide a layered defense in depth. The specific focus of BBAC is to analyze behaviors of actors and assess trustworthiness of information through machine learning. BBAC uses statistical anomaly detection techniques to make predictions about the intent of creating new TCP connections, issuing HTTP requests, sending emails, or making changes to documents. By focusing on behaviors that are nominally allowed by static access control policies but might look suspicious upon closer investigation, BBAC aims to detect targeted attacks that are currently going unnoticed for an extended amount of time, usually months before defenders are aware of cyber attacks.

Figure 1 shows a high-level diagram of the processing flow in BBAC together with the various data sets involved. As shown on the bottom, BBAC needs to ingest a large variety of data from real time feeds through a feature extraction process. During online use, this data will be used for classification purposes. However, for training purposes, BBAC needs to persist and manage training data sets. After parsing the raw observables, BBAC proceeds to go into a feature enrichment phase, where aggregate statistics are computed and information from multiple feeds is merged into a consistent representation. At this stage, BBAC needs to manage intermediate state required for more complex enrichment functions, e.g., calculating periodicity of events. For unsupervised learning, BBAC uses KMeans++ [2] clustering to group actors into clusters, which leads to a mapping function between actors and clusters. Within a cluster, BBAC performs supervised learning through a Support Vector Machine (SVM) [3] that is dedicated per cluster. Finally, action selection determines how to react to classification results obtained from the SVMs. Specific choices such as blocking requests, notifying administrators, or accepting requests are controlled via threshold policies.

2. Cyber Security Data Sets: Problems and Mitigation Strategies

As shown in Figure 1, BBAC is a data-intensive system with successful execution hinging on (a) access to a large amount of external data and (b) efficient management of internal data. Specifically, meaningful data sets are needed to develop and validate the accuracy, precision, and latency overhead of the BBAC algorithms and prototypes.

During development of BBAC, the following problems associated with cyber security data occurred and a number of mitigating strategies were devised. While there is no proven claim about coverage or even success associated with the strategies at this point, these strategies enabled BBAC research to proceed, both in terms of development and continuous validation.

Granularity mismatch. BBAC's analysis techniques work best with data that has a rich context and feature space. What is needed is a large amount of granular data to do statistical inference. Existing repositories, e.g., PREDICT [4], and deployed Host Based Intrusion Detection Systems (HIDSs), frequently

have two different problems. First, in the case of PREDICT, the repository has examples of labeled attack instances, but they are very narrow in scope (packet captures), leading to few features, and only representative of a small number of specific attacks. Second, in the case of HIDSs, access is limited to data that has been preprocessed by correlator nodes. Getting access to more granular information, e.g., involving access patterns of processes on end-systems, generally means installing software on end-systems or even recompiling applications (to map memory regions etc.), both of which raise practical concerns.

To address granularity issues, BBAC focuses its analysis on data that is easily observable without new software or modifying end systems. BBAC uses the following data sets:

- Bro packet sniffer logs both for extracting features about TCP connections and HTTP requests.
- Netflow data for TCP connections
- E-mail data from SMTP logs
- Chat data from XMPP logs
- Microtext data (from Twitter message archives)
- Page edit sequences (from Wikipedia page archives)
- Domain age and country IP information, as determined by information from Whois databases.

Table 1 lists the specific data sets used during the development and validation of BBAC.

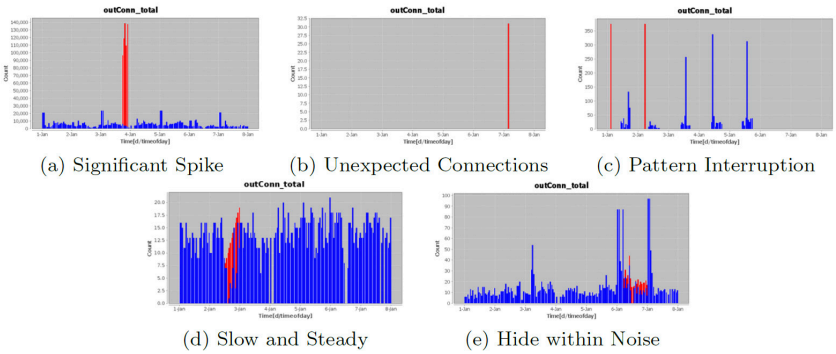

Figure 2. Simulated attack behaviors visible at the TCP layer. Blue lines indicate normal behavior and red lines indicate simulated attack behavior.

Lack of ground truth. To evaluate behaviors at the TCP level, the BBAC developers obtained a large data set of Bro [7] network traces from the BBN network. One immediate problem is that very little can be linked to actual confirmed attacks, leading to the situation in which a large part of the traffic needs to get labeled as unsuspicious.

To address the resulting lack of ground truth concerns, the BBAC developers simulated a number of different attack variants and observables based on how attackers are expected to exfiltrate data or spread attacks at the TCP level. This lead to development of the following randomized attacks:

- **Category 1:** Significant Spike. Significant consistent increase in outbound connections.
- **Category 2:** Unexpected Connections. These attacks show outbound activity where there never was any, e.g., a server that has never made outbound requests suddenly making outbound requests.
- **Category 3:** Pattern Interruption. Many hosts follow a regular pattern (e.g., servers fetching updates at regular intervals). The attacks cause interruptions in those patterns.
- **Category 4:** Slow and Steady. Slight increase over normal values, should still be detectable, though with lower accuracy.
- **Category 5:** Hide Within Noise. A highly sophisticated adversary might craft attacks that stay below trained thresholds. In the most extreme case, these attacks form a control case, as BBAC should not be able to detect them. However, it is likely that even the most skilled adversary will trigger Category 1-4 behaviors in a significant part of the hundreds of features observed by BBAC, raising detection accuracy to be significantly higher than random.

Type	Data Set
Network Flows	Connection summary data from Bro and Netflow data captured on the BBN network over the period of 1 month, plus simulated attacks
WHOIS	Domain name record information gathering by BBN for a set of 27839 domains.
HTTP Requests	HTTP request and response headers extracted by Bro on the BBN network collected over the period of 2 weeks, totaling ~17 million requests. In addition, a set of thousands of known bad URLs from PhishTank and URLblacklist. • http://www.phishtanbk.com • http://urlblacklist.com/?sec=download
Document Changes	Full page history dump of Simple English Wikipedia from 2/27/2012, containing 237,000 pages, 176,0000 users, and 3.1 million revisions. • http://dumps.wikimedia.org/simplewiki
Email	Publicly available email archives from Enron that were made public as part of US court proceedings [11] containing ~619k messages belonging to 158 users. • https://www.cs.cmu.edu/~enron/
Chat	Archives of message exchanges directly obtained from Twitter.

Table 1. Data Sets Used During BBAC Development and Evaluation

While the multi-category attack simulations help with attack realism, concerns remain that the BBACs defenses are limited by assumptions made about attacks. Going forward, it would be nice to get better ground truth, e.g., by tying the simulated attacks more directly to actual cyber events, such as the Stratfor Hack [8]. Another part of dealing with this problem is focusing on data that has proper ground truth, such as HTTP requests (for which blacklisted URLs exist) and Wikipedia edits (for which it is known whether the edits were reverted by the Wiki community or not).

Size of available data sets. Wikipedia [9] archives are one of the data sets BBAC uses for assessing document trustworthiness. Wikipedia has rich collaborative semantics attached to it and the archives not only contain the text of wiki pages but also maintain edit sequences over time and provide notions of ground truth, e.g., by including events where edits were reverted or users were banned. So, on first look, Wikipedia seems free of many of the problems described earlier: it is easy to access,

has good granularity, and observable ground truth. Aside from it only covering a very specific trust model and interaction style, the main problem is actually parsing and processing the vast amount of Wikipedia data. In addition, some important features such as the text that was changed between two edits, become computationally intractable quickly, as Wikipedia stores the full text across revisions, requiring extensive use of greatest common substring algorithms to find changes.

To avoid processing nightmares associated with standard Wikipedia, early development on BBAC switched to working with Simple English Wikipedia [10] instead. Simple English Wikipedia is a wiki comprised of a subset of pages from English Wikipedia written using simple vocabulary and grammar meant for those learning the English language. While maintaining content and structure that is similar to English Wikipedia, Simple English Wikipedia archives are significantly smaller in size, thereby increasing efficiency of early investigation of prediction algorithms.

Independence of data sets. Since BBAC performs analysis at multiple different system layers, it not only needs access to data from sensors at these layers but the data in each layer needs to be linked to the other layers to represent a consistent picture of observables.

To address the problem of independence between data sets, BBAC uses an approach for injecting malicious URLs into request streams of benign hosts. Known bad HTTP requests are retrieved from blacklists, and intelligently inserted into existing connections patterns. It is important to keep the ratio of normal vs. abnormal traffic roughly equal allowing the resulting classifier to make decisions both on known proper behavior as well as known improper behavior.

Furthermore, BBAC has a twofold mitigation strategy for establishing correlations across data sets: First, later versions of Bro already link data from TCP connections with data about HTTP requests through the use of session IDs. BBAC will reuse the session ID for establishing cross correlations in this case. Second, for cases in which multiple data streams are collected independently and no session ID exists, BBAC will create a session ID of its own by correlating information based on information about source and destination IP addresses and ports, together with size counts and timestamps.

Model overfitting. BBAC explores the space of machine learning parameters used by SVMs to find a combination of parameters that work best for a given data set. While this maximizes accuracy for the particular data set at hand, it also biases the model so that it might perform significantly worse on future data sets.

To prevent bias through model overfitting, it is advisable to keep a set of continuously updated data that has never been used for model training and parameter space exploration. These data sets can then be used to test the accuracy of the trained models while avoiding training bias.

3. Developing and Validating BBAC

To control the risks of developing new technology without a clear path to data access, the development approach to BBAC is based on the following key tenants that together aim to produce an innovative and applied prototype capability.

Spiral-based: The 3 year project was divided into 6 half-year long spirals, with technology demonstrations happening every 3 months. This rapid prototyping approach provided opportunity for frequent feedback and enabled the project to stay on track through a number of course corrections. Figure 3 shows the evolution of functionality across the current set of spirals, with the following key milestones:

1. Initial capability showing feasibility of using SVMs for the purpose of detecting suspicious TCP connections.
2. First graphical demonstration of analysis of TCP connections and HTTP requests, using features that were extracted ahead of time from real traffic collected at BBN as well as simulated attacks.
3. Enhanced demonstration, shown at the NSA Information Assurance Symposium 2012, including analysis of Wikipedia page edits as well as inclusion of Whois records to boost analysis accuracy.
4. Inclusion of KMeans clustering (non-supervised learning) to complement SVMs (supervised learning) to boost accuracy.
5. First support for mapping un-anticipated new actors to existing clusters through the use of decision trees to implement "intelligent clustering." Expansion of data sets to include stylometric analysis on email traffic. First version of using a cloud-based compute platform for classification of observables.
6. Expansion of observables to include chat messages and first scalability results associated with training.
7. Inclusion of analysis of blogs and first integrated demonstration showing feature extraction, training, and classification all being performed using the cloud framework.

Looking through the capabilities developed in those seven milestones, a number of points stand out that speak for the utility of the spiral-based approach. First, the BBAC team was able to give demonstrations on limited data sets very early on during the development (steps 1-3), which fostered discussions with a number of stakeholders, including transition partners and other researchers. Next, the focus in steps 4-5 switched to tackling more sophisticated use cases, such as handling of never before seen actors and dealing with environments where little training data is available. Finally, the operational realism of the prototype increased in steps 5-7 by including additional types of data and porting the implementation over to a cloud platform to address scalability requirements of expected deployments.

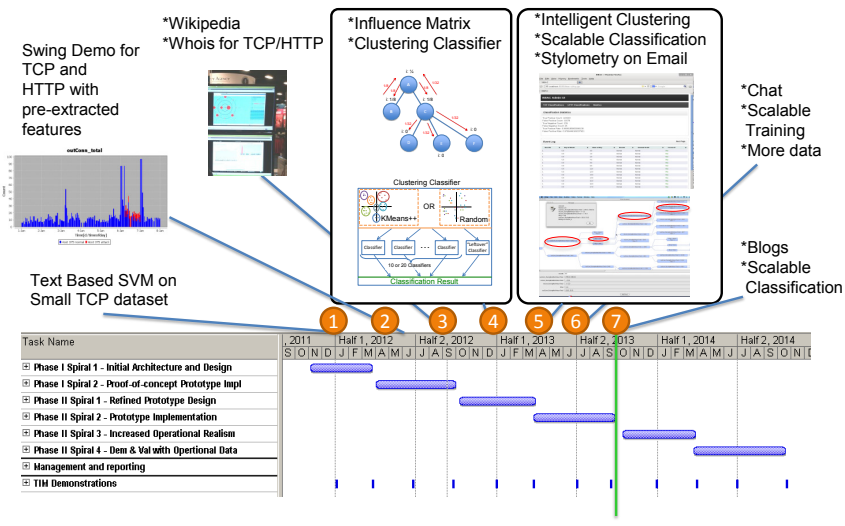

Figure 3. Spiral-based Agile Development Model for BBAC

Metric	Phase 1 Target	Phase 2 Target	Phase 1 Apr '12	Phase 1 Jun '12	Phase 1 Sep '12	Phase 2 Dec '12	Phase 2 Mar '13	Phase 2 Jun '13	Phase 2 Oct '13
Accuracy Characterization	TP >70%	TP>80%	76%	85%	91%	96%	96%	96%	96%
	FP < 1%	FP < 1%				0.45%	0.45%	0.45%	0.45%
Accuracy Attribution	TP>70%	TP>80%		89%	71%	76%	73%	73%	88%
	FP < 1%	FP < 1%							0.53%
Precision	>80%	>90%	76%	93%	97%	97%	98%	98%	98%
Latency	<1s	<100ms	623 ms	0.7 ms	0.7 ms	0.7 ms	0.7 ms	0.7 ms	0.7 ms
Overhead	<200%	<100%			70%	70%	70%	70%	70%
Scalability	~constant >2 nodes	~constant >10nodes						40 nodes	40 nodes
Security OWASP	100%	100%		100%	100%	100%	100%	100%	100%
Security 800-53	>25%	>60%		68%	68%	68%	68%	68%	68%
Flexibility	2	5	1 (TCP)	2 (+HTTP)	3 (+Wiki)	3	4 (+Email)	5(+Chat)	5
TRL	2	3	2	2	3	3	3	3	3

Table 2. Results from Continuous Assessment

Metric	Measured	Phase II Target	Status	Details
Accuracy	Correctness of characterization (CC) TP=% of attacks correctly identified FP=% of normal traffic incorrectly labeled as attack	TP ≥ 80% FP ≤ 1%	TP = 96% FP = 0.45%	TCP : [92.9%, 0.5%]* *reported as [TP,FP] HTTP: [99.6%, 0.9%]
	Correctness of attribution (CA)	TP ≥ 80% FP ≤ 1%	TP = 88% FP = 0.53%	Wiki: [76%,1%] Twitter: [96%, 0.18%] Email: [93%, 0.4%]
Precision	Positive Predictive Value (PPV) for CC PPV=# TP / (# TP + # FP)	≥ 90%	97%	TCP: 94.2%, HTTP: 99.1%
	PPV for CA	≥ 75%	99.6%	Wikipedia Edits*: 99%, Email: 99.99%, Twitter: 99.8%
Timeliness	Classification latency	< 100 ms	0.7 ms	TCP: m=0.74ms, s=0.52ms HTTP: m=19ms, s=5.4ms Wiki: m=1.3µs, s=30.2µs
	Overhead on inline access control decisions latencies	≤ 100%	70%	TCP(1ms): 70% , HTTP(100ms): 19%, Wiki (100ms): < 0.01%
Scalability	Training data set size / Number p of parallel processors	~constant p>10	~constant, p=40	
	Training data set size / Training time	~constant	~constant	
Security	Coverage over OWASP Top 10 threat categories	100%	100%	Assessed current system at 100%
	Adherence to applicable NIST-800-53 security controls (over 150)	30-60%	100%	Assessed current system at 100% 68 controls directly impacted by BBAC
Flexibility	Number of supported document and service types	≥5	5	TCP, HTTP, Wikipedia, Email, Chat
TRL	Technology Readiness Level (TRL)	3-4	3	Running in BBN cluster and delivered on VM

Table 3. Project Metrics

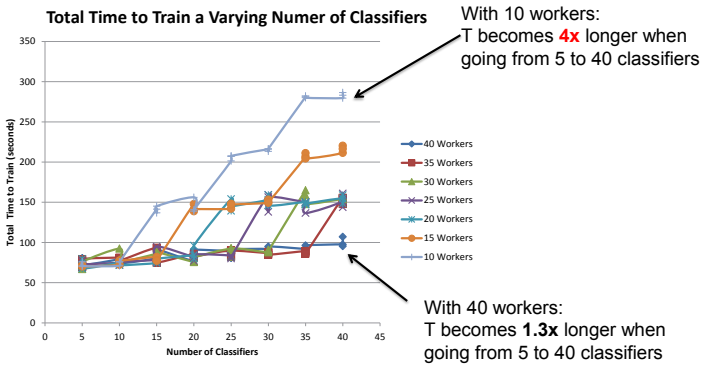

Figure 4. Scalability graph showing that the total training time for SVM classifiers (y axis) can be controlled by adding more worker nodes as the number of classifiers increases (x axis).

Metrics-driven: Progress is measured by tracking seven distinct quantitative metrics, as displayed in Table 3. Accuracy and Precision are determined via 10-fold cross validation. During n-fold cross validation, a data set is divided into n equal parts for n separate iterations. In each iteration, the model is trained on n-1 parts and tested on the remaining part. This guarantees that the data that is used for model construction is never used for model testing. Timeliness is determined by measuring classification and enforcement latencies. Scalability measures the ability of BBAC to just use more hardware to accommodate an increase in load. Security is measured by determining coverage over multiple threat and IA control models, while Flexibility measures the number of data sources BBAC can ingest. Finally, the Technology Readiness Level measures the maturity of the resulting software system.

Continuously Assessed: The set of metrics is assessed frequently throughout the development cycle, usually at the end of every spiral. Table 2 shows the progression of metric compliance across the project execution, with a number of interesting trends.

Attribution accuracy initially met the goal, but then stayed below the target threshold. This can be explained by (1) increasing the amount of Wikipedia data processed and (2) including stylometric analysis of email and chat as part of this metric.

Scalability was ignored during initial development, allowing development to focus on accuracy and precision metrics, but then addressed with refactoring of the functionality to a cloud framework. Figure 4 shows how BBAC's processing scales with increased load by distributing processing over a dynamically assignable set of processing nodes. The figure shows the total time to train a varying number of classifiers. Each colored line represents a different configuration with the number of worker nodes ranging between 10 and 40. Looking at the curve with 10 workers, training time of 10 classifiers takes about 70 seconds, while training of 40 classifiers takes about 280 seconds (~ 4 times as long). By adding more worker nodes, training time of 40 classifiers can be brought down to 100 seconds through parallelized processing across cluster nodes.

Conclusion

Development and validation of statistical cyber defenses needs a well-labeled, appropriately sized, and readily available amount of relevant data to make innovative progress, yet too little of such data sets are available today. This article describes an initial set of data challenges and solutions from the work on BBAC, and describes how agile project management techniques helped deliver innovative technology in a difficult to work in data-intensive environment. Going forward, we intend to provide a list of requirements that will help data providers and clearinghouses create and maintain data sets that are more effective in driving R&D development of the next generation of cyber security technologies.

Disclaimers:

This work was sponsored by the Air Force Research Laboratory (AFRL). Distribution A. Approved for public release; distribution unlimited (Case Number 88ABW-2013-4318).

Acknowledgments:

We acknowledge the support of various team members that helped address data problems on the BBAC effort, including John Benner of Booz Allen Hamilton; Andrew McDonald and Jeffrey Segall of Drexel University; and Jeffrey Cleveland of Raytheon BBN Technologies.

REFERENCES

1. NIST SP 800-162: Guide to Attribute Based Access Control (ABAC) Definition and Considerations (Draft), <http://csrc.nist.gov/publications/drafts/800-162/sp800_162_draft.pdf>
2. Arthur, David, and Sergei Vassilvitskii. "k-means++: The advantages of careful seeding." Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2007.
3. Wang, Lipo, ed. Support Vector Machines: theory and applications. Vol. 177. Springer, 2005.
4. Protected Repository for the Defense of Infrastructure Against Cyber Threats (PREDICT), <https://www.predict.org/>
5. B.Claise, "Cisco Systems NetFlow Services Export Version 9", October 2004, RFC 3954, <http://tools.ietf.org/html/rfc3954>
6. L. Daigle, "WHOIS Protocol Specification", September 2004, RFC 3912, <http://tools.ietf.org/html/rfc3912>
7. The BRO Network Security Monitor, <http://bro-ids.org/>
8. Strafor Hack, <http://www.stratfor.com/weekly/hack-stratfor>
9. Wikipedia – The free encyclopedia, <http://www.wikipedia.org/>
10. Simple English Wikipedia, <http://simple.wikipedia.org/wiki/Main_Page>
11. Klimt, Bryan, and Yiming Yang. "Introducing the Enron Corpus." CEAS. 2004.

ABOUT THE AUTHORS

Michael Atighetchi is a Senior Scientist in the Information and Knowledge Technology business unit at BBN Technologies. Since Mr. Atighetchi joined BBN more than 13 years ago, he has contributed to research and technology development in the areas of cross domain information sharing, adaptive QoS middleware, cognitive reasoning in cyber defense, system survivability and security verification through red team testing. He has been a key technical contributor to several DARPA- and USAF-sponsored research projects. Mr. Atighetchi has published over 60 technical papers in peer-reviewed journals and conferences and is Senior Member of the IEEE.

Raytheon BBN Technologies
10 Moulton Street
Cambridge, MA 02138
Phone: 617-873-1679
E-mail: matighet@bbn.com

ABOUT THE AUTHORS

Michael Jay Mayhew, a Senior Computer Scientist, has been with AFRL Information Directorate for the past 17 years, 10 of those years as a Federal Civilian. As the Program Manager of the Cross-Domain Innovation & Science (CDIS) group, Mr. Mayhew leads a team of research engineers in finding and developing new cross-domain technologies and maturing those technologies for integration within existing cross-domain products. Mr. Mayhew is a frequent presenter at worldwide program and technical conferences each year and is recognized as a subject matter expert in the area of cross-domain technology.

Air Force Research Laboratory
525 Brooks Road
Rome, NY 13441
Phone: 315-330-2898
E-mail: michael.mayhew@rl.af.mil

Dr. Rachel Greenstadt is an Assistant Professor of Computer Science at Drexel University. Dr. Greenstadt's research centers on the privacy and security properties of multi-agent systems and the economics of electronic privacy and information security. Her lab -- the Privacy, Security, and Automation Laboratory (PSAL) -- focuses on designing more trustworthy intelligent systems that act autonomously and with integrity, so that they can be trusted with important data and decisions. The lab takes a highly interdisciplinary approach to this research, incorporating ideas from artificial intelligence, psychology, economics, data privacy, and system security. However, a common thread of this work has been studying information flow, trustworthiness, and control. Recently, much of the work has focused on using machine learning to better understand textual communication.

Drexel University
University Crossings 140
Philadelphia, PA 19104
E-mail: greenie@cs.drexel.edu

Dr. Aaron Adler is a Scientist in the Information and Knowledge Technologies business unit at BBN Technologies. Dr. Adler joined BBN in 2009 after finishing his Ph.D. thesis at MIT. He has contributed to research and technology development in several areas including: cyber defense, synthetic biology, advanced design tools, AI learning, and quantum compilers. He is currently a co-PI for a DARPA-funded project and has been a key technical contributor to several other DARPA- and USAF-funded research projects.

Raytheon BBN Technologies
10 Moulton Street
Cambridge, MA 02138
Phone: 617-873-3517
E-mail: aadler@bbn.com