REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
24-05-2017	Final Report	31-May-2011 - 30-Mar-2018

4. TITLE AND SUBTITLE
Final Report: Proactive Detection of Insider Threats with Graph Analysis and Learning (PRODIGAL)

5a. CONTRACT NUMBER

5b. GRANT NUMBER
W911NF-11-C-0088

5c. PROGRAM ELEMENT NUMBER
1M30BM

6. AUTHORS
Matthew G Reardon, Henry G Goldberg, Brian J Phillips

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
Leidos, Inc. DBA Surveillance and Reconna
11951 Freedom Drive

Reston, VA 20190 -0000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
60133-NS-DRP.46

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
Leidos developed and operated a prototype system (PRODIGAL) as a testbed for exploring a range of insider threat detection and analysis methods. The data and test environment, system components, and the core method of unsupervised detection of insider threat leads are presented to benefit others working in the insider threat domain. We discuss a set of experiments evaluating the prototype's ability to detect both known and unknown malicious insider behaviors. The experimental results show the ability to detect a large variety of insider threat scenario instances embedded in real data with no prior knowledge of what scenarios are present or when they occur. We

15. SUBJECT TERMS
insider threat, anomaly detection, explanation, unsupervised ensembles, graph analysis, user activity monitoring

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Ted Senator
UU	UU	UU	UU		19b. TELEPHONE NUMBER 703-248-7728

Report Title

Final Report: Proactive Detection of Insider Threats with Graph Analysis and Learning (PRODIGAL)

ABSTRACT

Leidos developed and operated a prototype system (PRODIGAL) as a testbed for exploring a range of insider threat detection and analysis methods. The data and test environment, system components, and the core method of unsupervised detection of insider threat leads are presented to benefit others working in the insider threat domain. We discuss a set of experiments evaluating the prototype's ability to detect both known and unknown malicious insider behaviors. The experimental results show the ability to detect a large variety of insider threat scenario instances embedded in real data with no prior knowledge of what scenarios are present or when they occur. We report on an ensemble-based, unsupervised technique for detecting potential insider threat instances. When run over 16 months of real monitored computer usage activity augmented with independently developed and unknown but realistic, insider threat scenarios, this technique robustly achieves results within five percent of the best individual detectors identified after the fact. We discuss factors that contribute to the success of the ensemble method, such as the number and variety of unsupervised detectors and the use of domain knowledge encoded in detectors designed for specific activity patterns.

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received		Paper
05/24/2017	43	Chau, D.H., Faloutsos, C.. Case study on fraud detection using social-network analysis., Encyclopedia of Social Networks and Mining, ( ): 547. doi:
		1,042,205.00
TOTAL:	**1**	

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received		Paper
TOTAL:		

Number of Papers published in non peer-reviewed journals:

(c) Presentations

Number of Presentations: 0.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received		Paper
05/19/2017	4	Akoglu, L., Chau, D. H., Kang, U., Koutra, D., and Faloutsos, C. Large Graph Mining System for Patterns, Anomalies & Visualization, 16th Pacific-Asia Conference, PAKDD 2012. 29-MAY-12, Kuala Lumpur, Malaysia. : ,
05/19/2017	3	Akoglu, L., Chandy, R., and Faloutsos, C.. Opinion Fraud Detection in Online Reviews using Network Effects, 7th Annual International Conference on Weblogs and Social Media. 08-JUL-13, Cambridge, Massachusetts. : ,
05/23/2017	5	Akoglu, L., Vreeken, J., Tong, H., Chau, D.H., and Tatti, N.. Mining Connection Pathways for Marked Nodes in Large Graphs, Society for Industrial and Applied Mathematics 2013 SIAM Conference on Data Mining (SDM13). 03-MAY-13, Austin, TX. : ,
05/23/2017	41	Young, W. T., Memory, A., Goldberg, H. G., and Ted, E.. Detecting unknown insider threat scenarios, Security and Privacy Workshops (SPW), 2014 IEEE. 18-MAY-14, San Jose, CA. : ,
05/23/2017	40	Young, W T. et. al. Use of Domain Knowledge to Detect Insider Threats in Computer Activities, the Workshop on Research for Insider Threat, IEEE CS Security and Privacy Workshops. 24-MAY-13, San Francisco, CA. : ,
05/23/2017	39	Wallnau, K., et. al.. Simulating malicious insiders in real host-monitored user data, 7th USENIX conference on Cyber Security Experimentation and Test. 19-AUG-14, San Diego, CA. : ,
05/23/2017	38	Stolper, C., Kahng, M., Lin, Z., Foerster, F., Goel, A., Stasko, J., and Chau, D. H.. GLO-STIX: Graph-Level Operations for Specifying Techniques and Interactive eXploration, IEEE InfoVis 2014. 10-NOV-14, Paris, France. : ,
05/23/2017	37	Stolper, C., Foerster, F., Kahng, M., Lin, Z., Goel, A., Stasko, J., Chau, D. H.. GLOs: Graph-Level Operations for Exploratory Network Visualization, CHI 2014. 27-APR-14, Toronto, Canada. : ,
05/23/2017	36	Stolper, C., Foerster, F., Kahng, M., Lin, Z., Goel, A., Stasko, J., Chau, D. H.. GLOs: Graph-Level Operations for Exploratory Network Visualization, CHI 2014. 27-APR-14, Toronto, Canada. : ,
05/23/2017	35	Siddiqui, A., Fern, A., Dietterich, T. G., & Wong, W.. Sequential Feature Explanations for Anomaly Detection, KDD 2015 Workshop on Outlier Definition, Detection, and Description. 11-AUG-15, Sydney, Australia. : ,
05/23/2017	34	Senator, T., et al.. Detecting Insider Threats in a Real Corporate Database of Computer Usage Activity, ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD). 12-AUG-13, Chicago, IL. : ,
05/23/2017	33	Pienta, R., Abello, J., Kahng, M., Chau, D. H.. Scalable Graph Exploration and Visualization: Sensemaking Challenges and Opportunities, International Conference on Big Data and Smart Computing (BigComp). 10-FEB-15, Jeju Island, Korea. : ,
05/23/2017	32	Papalexakis, E.E., Akoglu, L., and Ienco, D.. Do more Views of a Graph help? Community Detection and Clustering in Multi-Graphs, 16th International Conference on Information Fusion. 10-JUL-13, Istanbul, Turkey. : ,

05/23/2017 31 Memory, A., Senator, T.. Towards Robust Anomaly Detection Ensembles using Explanations, KDD 2015 Workshop on Outlier Definition, Detection, and Description. 11-AUG-15, Sydney, Australia. : ,

05/23/2017 30 Memory, A., Goldberg, H. G., & Senator, T. E.. Context-aware insider threat detection, Twenty-Seventh AAAI Conference on Artificial Intelligence. 15-JUL-13, Bellevue, WA. : ,

05/23/2017 29 McColl, R., Green, O., Bader, D.. A New Parallel Algorithm for Connected Components in Dynamic Graphs, IEEE International Conference on High Performance Computing. 19-DEC-13, Bengaluru (Bangalore), Karnataka, India. : ,

05/23/2017 28 Maier, M., Marazopoulou, K., Arbour , D., Jensen, D.. A sound and complete algorithm for learning causal models from relational data, 29th Conference on Uncertainty in Artificial Intelligence. 13-JUL-13, Bellevue, WAS. : ,

05/23/2017 27 Lin, Z., Chau, D. H.. Interactive Multi-resolution Exploration of Million Node Graphs, IEEE VIS 2013. 14-OCT-13, Atlanta, GA. : ,

05/23/2017 26 Lin, Z., Cao, N., Tong, H., Wang, F., Kang, U, and Chau, D. H.. Demonstrating Interactive Multi-resolution Large Graph Exploration, ICDM 2013. 08-DEC-13, Dallas, TX. : ,

05/23/2017 25 Lin, Y., Raza, A. A., Lee, J-Y., Koutra, D., Rosenfeld, R., Faloutsos, C.. Influence Propagation: Patterns, Model and a Case Study, Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD). 14-MAY-14, Tainan, Taiwan. : ,

05/23/2017 24 Lee, J.-Y., Kang, U., Koutra, D., and Faloutsos, C.. Fast anomaly detection despite the duplicates, 22nd International Conference on World Wide Web. 14-MAY-13, Rio de Janeiro, Brazil. : ,

05/23/2017 23 Koutra, D., Vogelstein, J.T., Faloutsos, C.. DELTACON: A Principled Massive-Graph Similarity Function, Society for Industrial and Applied Mathematics 2013 SIAM Conference on Data Mining (SDM13). 03-MAY-13, Austin, TX. : ,

05/23/2017 22 Koutra, D., Papalexakis, E. E., Faloutsos, C.. TENSORSPLAT: Spotting Latent Anomalies in Time, 16th PanHellenic Confernece on Informatics with International Participation. 06-OCT-12, Piraeus, Greece. : ,

05/23/2017 21 Koutra, D., Koutras, V., Prakash, B.A., and Faloutsos, C.. Patterns among Competing Tasks Frequencies: Super-Linearities, and the Almond-DG Model, 17th Pacific-Asia Conference, PAKKDD 2013. 15-APR-13, Gold Coast, Australia. : ,

05/23/2017 20 Koutra, D., Kang, U., Vreeken, J., & Faloutsos, C.. VOG: Summarizing and Understanding Large Graphs, SDM 2014. 25-APR-14, Philadelphia, PA. : ,

05/23/2017 19 Kang, U, Lee, J-Y., Koutra, D. and Faloutsos, C. Net-Ray: Visualizing and Mining Billion-Scale Graphs, Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD). 14-MAY-14, Tainan, Taiwan. : ,

05/23/2017 18 Jiang, M., Cui, P., Beutel, A., Faloutsos C.and Yang S. CatchSync: Catching Synchronized Behavior in Large Directed Graphs, KDD 2014. 25-AUG-14, New York City, NY. : ,

05/23/2017 17 Günnemann, N. and Faloutsos, C. Robust Multivariate Autoregression for Anomaly Detection in Dynamic Product Ratings, International World Wide Web Conference (WWW). 08-APR-14, Seoul, Korea. : ,

05/23/2017 16 Green, O., Bader, D. Faster Betweenness Centrality Based on Data Structure Experimentation, 13th International Conference on Computational Science (ICCS). 06-JUN-13, Barcelona, Spain. : ,

05/23/2017 15 Goldberg, H. G., Young, W. T., Reardon, Matthew G., Phillips, Brian J., Senator, T.E. Insider Threat Detection in PRODIGAL,
50th Hawaii International Conference on System Sciences. 05-JAN-17, Waikoloa, HI. : ,

05/23/2017 14 Goldberg, H. G., Young, W. T., Memory, A. C., Senator, T.E. Explaining and Aggregating Anomalies to Detect Insider Threats,
49th Hawaii International Conference on System Sciences. 27-JAN-16, Kauai, HI. : ,

05/23/2017 13 Friedland,L., Lavine, M., Jensen, D. Copy or coincidence? A model for detecting social influence and duplication events,
30th International Conference on Machine Learning. 17-JUN-13, Atlanta, GA. : ,

05/23/2017 12 Friedland, L., Gentzel, A., and Jensen, D. Classifier-adjusted density estimation for anomaly detection and one-class classification,
2014 SIAM International Conference on Data Mining. 25-APR-14, Philadelphia, PA. : ,

05/23/2017 11 Emmott A., Das, S., Dietterich, T., Fern, A. and Wong, W-K. Systematic Construction of Anomaly Detection Benchmarks from Real Data,
SIGKDD Workshop on Outlier Detection and Description. 12-AUG-13, Chicago, IL. : ,

05/23/2017 10 Bettadapura, V., Thomaz E., Parnami A, Abowd, G., and Essa, I.. Leveraging Context to Support Automated Food Recognition in Restaurants,
IEEE Winter Conference on Applications of Computer Vision (WACV). 07-JAN-15, Waikoloa Beach, HI. : ,

05/23/2017 9 Bettadapura, V., Schindler, G., Plötz T., and Essa, I.. Augmenting Bag-of-Words: Data-Driven Discovery of Temporal and Structural Information for Activity Recognition,
IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 24-JUN-13, Portland, OR. : ,

05/23/2017 8 Berlingerio, M., Koutra, D., Eliassi-Rad, T., and Faloutsos, C. NetSimile: A Scalable Approach to Size-Independent Network Similarity,
Workshop on Information in Networks 2012. 29-SEP-12, New York, NY. : ,

05/23/2017 7 Arbour, D., Atwood, J., Jensen, D. Agglomerative clustering of bagged data using joint distributions,
ICML Workshop on Structured Learning: Inferring Graphs from Structured and Unstructured Inputs. 13-JUL-13, Atlanta, GA. : ,

05/23/2017 6 Araujo, M., Papadimitriou, S., Günnemann, S., Faloutsos, S., Basu, P., Swami, A., Papalexakis , E., and Koutra, D.. Com2: Fast Automatic Discovery of Temporal ('Comet') Communities Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD),
Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD). 14-MAY-13, Tainan, Taiwan. : ,

05/24/2017 42 Ankerst, Mihael, et al.. OPTICS: Ordering Points To Identify the Clustering Structure,
ACM SIGMOD'99 Int. Conf. on Management of Data. 31-MAY-99, Philadelphia, PA. : ,

05/24/2017 44 Lee, J.-Y., Kang, U., Koutra, D., and Faloutsos, C.. Fast anomaly detection despite the duplicates,
22nd International Conference on World Wide Web. 13-MAY-13, Rio de Janeiro, Brazil. : ,

05/24/2017 45 Zakrzewska, A., Bader, D.. Measuring the Sensitivity of Graph Metrics to Missing Data,
10th International Conference on Parallel Processing and Applied Mathematics. 08-SEP-13, Warsaw, Poland. : ,

TOTAL: 42

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received Paper

TOTAL:

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received Paper

TOTAL:

Number of Manuscripts:

Books

Received Book

TOTAL:

Received Book Chapter

TOTAL:

Patents Submitted

Patents Awarded

Awards

Best Paper Award:
Young, W. T., Memory, A., Goldberg, H. G., and Senator Ted, E. (2014, May). Detecting unknown insider threat scenarios. 2014 IEEE Security and Privacy Workshops (SPW), IEEE (pp. 277-288). IEEE.

Best Paper Award:
STINGER: High Performance Data Structure for Streaming Graphs, D. Ediger, R. McColl, J. Riedy, and D.A. Bader, 2012 IEEE High Performance Extreme Computing Conference (HPEC). IEEE.

Distinguished Professor:
Thomas Dietterich, Oregon State University, 2013.

Graduate Students

NAME	PERCENT_SUPPORTED	DISCIPLINE
James Atwood - UMass	76	Computer and Computational Sciences
Elisabeth Baseman - UMass	26	Computer and Computational Sciences
Lisa Friedland - UMass	97	Computer and Computational Sciences
Amanda Gentzel - UMass	65	Computer and Computational Sciences
Daniel Garant - UMass	75	Computer and Computational Sciences
Marc Maier - UMass	81	Computer and Computational Sciences
Katerina Marazopoulou - UMass	39	Computer and Computational Sciences
Huseyin Oktay - UMass	78	Computer and Computational Sciences
Matthew Rattigan - UMass	42	Computer and Computational Sciences
Brian Taylor - UMass	63	Computer and Computational Sciences
Danai Koutra - CMU	75	Computer and Computational Sciences
Polo Chau - CMU	75	Computer and Computational Sciences
Jay-Yoon Lee - CMU	75	Computer and Computational Sciences
Shubhomoy Das - OSU	100	Computer and Computational Sciences
Junyuan Lin - OSU	100	Computer and Computational Science and Electrical E
Joe Selman - OSU	100	Computer and Computational Sciences
Andrew Emmott - OSU	100	Computer and Computational Sciences
Md. Tahmid-un Nabi	49	Computer and Computational Sciences
Tadesse Zemicheal - OSU	100	Computer and Computational Sciences
Vinay Bettadapura - GT	27	Computer and Computational Sciences
Aaskash Goel - GT	9	Computer and Computational Science and Electrical E
Oded Green - GT	75	Computer and Computational Science and Electrical E
Daniel Henderson - GT	9	Computer and Computational Science and Electrical E
Oguz Kaya - GT	38	Computer and Computational Science and Electrical E
Elias Khalil - GT	9	Computer and Computational Science and Electrical E
Jiajia Li - GT	10	Computer and Computational Science and Electrical E
Robert McColl - GT	7	Computer and Computational Science and Electrical E
Acar Tamersoy - GT	18	Computer and Computational Sciences
Anita Zakrewska - GT	37	Computer and Computational Science and Electrical E
FTE Equivalent:	**16.55**	
Total Number:	**29**	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
Michael Shindler	1.00
Daniel Corkill	0.50
Jussara Kofuji	0.09
FTE Equivalent:	**1.59**
Total Number:	**3**

Names of Faculty Supported

NAME	PERCENT_SUPPORTED	National Academy Member
David Bader	0.15	
David Jensen	0.15	
Thomas Dietterich	0.15	
Christos Faloutsos	0.10	
Richard Boyd - GTRI	0.25	
Erica Briscoe - GTRI	0.25	
Irfan Essa	0.10	
Edmond Chow	0.10	
Polo Chau	0.10	
Weng-Keen Wong	0.10	
Alan Fern	0.10	
FTE Equivalent:	**1.55**	
Total Number:	**11**	

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED	DISCIPLINE
Kayla Loony - OSU	25	Computer and Computational Sciences
Samarth Agarwal - GT	9	Computer and Computational Sciences
Lin Zhiyuan - GT	2	Computer and Computational Sciences
FTE Equivalent:	**0.36**	
Total Number:	**3**	

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period: 3.00

The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:..... 3.00

The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:..... 1.00

Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):..... 2.00

Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:..... 0.00

The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense 0.00

The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:..... 1.00

Names of Personnel receiving masters degrees

NAME	
James Atwood - UMass	
Elizabeth Baseman - UMass	
Daniel Garant - UMass	
Amanda Gentzel - UMass	
Junyuan Lin - OSU	
Oguz Kaya - GT	
Anita Zakrweska - GT	
Aaskash Goel - GT	
Jiajia Li - GT	
Total Number:	**9**

Names of personnel receiving PHDs

NAME	
Lisa Friedland - UMass	
Marc Maier - UMass	
Matthew Rattigan - UMass	
Brian Taylor - UMass	
Danai Koutra - CMU	
Oded Green - GT	
Vinay Bettadapura - GT	
Total Number:	**7**

Names of other research staff

NAME	PERCENT_SUPPORTED
Ted Senator - Leidos	0.35
Henry Goldberg - Leidos	0.60
Alex Memory - Leidos	0.75
William Young - Leidos	0.40
Brian Phillips - Leidos	0.75
John Rotter - Leidos	0.75
Matthew Reardon - Leidos	0.20
Dan Hirpara - Leidos	0.25
David Lippert - Leidos	0.25
Brad Rees - Leidos	0.50
FTE Equivalent:	**4.80**
Total Number:	**10**

Sub Contractors (DD882)

1 a. Georgia Tech Research Institute

1 b. 410 10th Street
North Buillding
Atlanta GA 303320807

Sub Contractor Numbers (c):
Patent Clause Number (d-1):
Patent Date (d-2):
Work Description (e):
Sub Contract Award Date (f-1):
Sub Contract Est Completion Date(f-2):

---

1 a. Georgia Tech Research Institute

1 b. 410 10th Street
North Buillding
Atlanta GA 303320807

Sub Contractor Numbers (c):
Patent Clause Number (d-1):
Patent Date (d-2):
Work Description (e):
Sub Contract Award Date (f-1):
Sub Contract Est Completion Date(f-2):

---

1 a. Oregon State University

1 b. Office of Sponsored Programs
B308 Kerr Administration Building
Corvallis OR 973312140

Sub Contractor Numbers (c):
Patent Clause Number (d-1):
Patent Date (d-2):
Work Description (e):
Sub Contract Award Date (f-1):
Sub Contract Est Completion Date(f-2):

---

1 a. Oregon State University

1 b. Research Office
312 Kerr Administration
Corvallis OR 973312140

Sub Contractor Numbers (c):
Patent Clause Number (d-1):
Patent Date (d-2):
Work Description (e):
Sub Contract Award Date (f-1):
Sub Contract Est Completion Date(f-2):

1 a. University of Massachusetts - Amherst

1 b. 70 Butterfield Terrace

Amherst MA 010039242

Sub Contractor Numbers (c):

Patent Clause Number (d-1):

Patent Date (d-2):

Work Description (e):

Sub Contract Award Date (f-1):

Sub Contract Est Completion Date(f-2):

---

1 a. University of Massachusetts - Amherst

1 b. Research Administration Building
70 Butterfield Terrace

Amherst MA 010039242

Sub Contractor Numbers (c):

Patent Clause Number (d-1):

Patent Date (d-2):

Work Description (e):

Sub Contract Award Date (f-1):

Sub Contract Est Completion Date(f-2):

---

1 a. Carnegie Mellon University

1 b. 406 Mellon University
5000 Forbes Avenue

Pittsburgh PA 15213

Sub Contractor Numbers (c):

Patent Clause Number (d-1):

Patent Date (d-2):

Work Description (e):

Sub Contract Award Date (f-1):

Sub Contract Est Completion Date(f-2):

Inventions (DD882)

Scientific Progress

Executive Summary

The Proactive Discovery of Insider Threats Using Graph Analysis and Learning (PRODIGAL) team, comprised of Leidos Advanced Solutions Group and our university partners, Georgia Tech Research Institute / Georgia Institute of Technology, Oregon State University, University of Massachusetts, and Carnegie Mellon University, lead a research and development program focused on anomaly detection and explanation applied to the insider threat domain. This paper reports on our fundamental anomaly detection and explanation research, and the development and experimentation with our prototype PRODIGAL insider threat detection system as a testbed for exploring a range of detection and analysis methods. The data and test environment, system components, and the core method of unsupervised detection of insider threat leads are presented to document this work and benefit others working in the insider threat domain. We also discuss a core set of experiments evaluating the prototype's ability to detect both known and unknown malicious insider behaviors. The experimental results show the ability to detect a large variety of insider threat scenario instances embedded in real data with no prior knowledge of what scenarios are present or when they occur.

Research activities of the PRODIGAL team under the ADAMS program fall into four major categories:
- Fundamental research into anomaly detection algorithms, models, and feature selection and normalization techniques;
- Domain-specific research to provide meaningful indicators and scenarios for detection and explanation of potential insider threat;
- Design and implementation of an experimental ensemble-based framework for research into insider threat detection, including all required engineering and data exploration over the ADAMS program database of user activities; and
- Experimentation over 16 months of program data seeded with dozens of inserted targets to compare, contrast, and combine detection methods.

Fundamental Anomaly Detection Research: Our university-based team members developed innovative algorithms for structural anomaly detection. Several employ statistical models and operate over populations of feature vectors in an unsupervised fashion to estimate the likelihood that a member was created by the same process that created the bulk of the population (e.g. RPAD, RIDE, RDE, Cross Prediction, Ensemble GMM). Significant effort was devoted to feature selection and normalization techniques, which have proven highly successful on the ADAMS program data. Temporal models were developed to find anomalous sequences of actions (Temporal-based Anomaly Detection, Vector Space Models). These approaches represent user activities as sequential patterns, possibly at multiple levels of granularity, and identify unlikely sequences. Additional research focused on leveraging the human analyst by providing methods for exploring huge graphs or high dimensional spaces for anomalous or unexplained behaviors (Oddball, Apolo). Finally, significant research efforts addressed ways of exploiting the graph structures inherent in large collections of human-computer transactions (STINGER, Community Detection, Seed Set Expansion).

Domain-Specific Insider Threat Detection Research: Based on prior research into insider threat, PRODIGAL team members developed a large set of features based on domain-specific concepts such as activity type and their use in known threat scenarios. These include measures of computer user activity levels, statistical outlier measures of these activities, and ratios of activities designed to uncover unusual behavior. Features were implemented within the PRODIGAL framework for use in statistical outlier detection as well as by other structural anomaly detection algorithms. Complex pattern detectors for known threat scenarios were also implemented using a novel, graphical Anomaly Detection Language (Fraudster, Sabotage, IP Theft, Ambitious Leader).

Ensemble-Based Research Framework: Leidos developed an ensemble-based, unsupervised technique for detecting potential insider threat instances. The PRODIGAL framework ingests computer usage activity observations, computes features, operates over 100 anomaly detectors, and combines the results from the multiple detectors using an ensemble technique that provides the analyst with a single score for each user, for each day, serving as a starting point for further investigation as illustrated in Figure 1.
Figure 1 Illustration of how PRODIGAL combines the results from multiple detectors with an innovative ensemble technique that provides the analyst a single anomaly score.
We have also begun incorporating explanation capabilities with the ensemble approach so that underlying reasons for detection from individual detectors can be combined in the final result presented to analysts. These explanations are made available to the analysts in a user interface, with preliminary methods for incorporating user feedback within the PRODIGAL dashboard based upon the rich set of available features and anomaly detector scores. The report further describes the architecture of the prototype system, the environment in which we conducted these experiments and potential transition environments.

Insider Threat Detection Experimentation: When run over 16 months of real monitored computer usage activity augmented with independently developed and unknown but realistic, insider threat scenarios, this technique robustly achieves results within five percent of the best individual detectors identified after the fact. We discuss factors that contribute to the success of the ensemble method, such as the number and variety of unsupervised detectors and the use of prior knowledge encoded in detectors designed for specific activity patterns. This experimentation represents more than 1800 runs of detectors, varying by type of user activity, structural model (vector, sequence, graph), baseline population (peer group, shared resources, shared communications), as well as over time. This variety supports evaluation of robustness as well as sensitivity of detection

methods.

Figure 2 illustrates how these activities have interacted and supported one another over the course of the ADAMS program.

See attachments for remainder of document.

Technology Transfer

DARPA Anomaly Detection At Multiple Scales (ADAMS)

Proactive Discovery of Insider Threats Using Graph Analysis and Learning (PRODIGAL)

Final Report

Reporting Period: May 2011 – November 2016

Contracted By:

US Army Research, Development and Engineering Command (RDECOM)

Issued by USA/ARO Under Contract No. **W911NF-11-C-0088**

Leidos Advanced Solutions Group
4001 N Fairfax Drive, Suite 800
Arlington, Virginia 22203

Contents

Introduction **3**

Executive Summary **4**

Fundamental Research **7**
1.1. Georgia Tech Research Institute (GTRI) Algorithm Research Activities 7
1.1.1. Research in scalable algorithms for network analysis (GTRI SOW 1.1.1) 7
1.1.2. Research in massive temporal and visual dynamic graph analytics (GTRI SOW 2.1.1) 11
1.2. Carnegie Mellon University Algorithm Research Activities 18
1.2.1. Research in models of graph evolution (CMU SOW 1.1.2) 18
1.2.2. Research in graph summarization and understanding (CMU SOW 2.1.2) 20
1.3. Oregon State University Algorithm Research Activities 22
1.3.1. Research in for anomaly detection (OSU SOW 1.1.3) 22
1.3.2. Research in anomaly explanations 29
1.4. University of Massachusetts Algorithm Research Activities 30
1.4.1. Research in joint probability and causal reasoning models (UMass SOW 1.1.4) 31
1.4.2. Research in causal dependence in complex domains (UMass SOW 2.1.4) 34

Insider Threat Detection Research & Results **38**
1.5. Counter-Intelligence Domain Analysis (Leidos SOW 1.2.1) 38
1.6. Content Analysis Research (Leidos SOW 1.3) 45
1.7. Initial Experiments with Surrogate Data (All SOW 1.3) 46
1.8. Experimentation with Live Data using the PRODIGAL Framework (All SOW 1.3; SOW 2.2) 47
1.8.1. Test Data and Red Team Scenarios 47
1.8.2. A Diverse Suite of Anomaly Detectors 51
1.8.3. Measuring Detector Performance 54
1.9. Experiments with Detector Diversity 57
1.10. Temporal Aggregation Experiments (Leidos SOW 2.2.2) 59
1.10.1. Background 59
1.10.2. Methodology 60
1.10.3. Metrics and evaluation 61
1.10.4. Follow-on work 65

Algorithm / Evidence Combination and Explanation (Leidos SOW 2.2.2) **66**
1.11. Exploiting a Diverse Ensemble of Detectors 66
1.11.1. A Language for Combining Detectors and Specifying User Contexts 66
1.11.2. Unsupervised Ensemble-Based Anomaly Detection 69
1.11.3. A New Ensemble Using Explanations 77
1.12. Explanations 79

1.12.1. A Common, Understandable Representation for Anomaly Explanations 79
1.12.2. Evaluating Explanations (Leidos SOW 2.2.1) 81

System Development & Integration (Leidos SOW 1.2.2) 83
1.13. ADAMS Program Environment 83
1.14. ADAMS Environment Extensions 84
1.15. Component Architecture 85
1.16. Executing PRODIGAL 87
1.17. Transitioning PRODIGAL to a real-world enterprise 87

Supporting the End User (Leidos SOW 2.2.1) 89
1.18. The Analyst Interface 89
1.18.1. Presenting Ensemble Scores in Context 89
1.19. Explaining Anomaly Scores 91

Transition Preparation and Recommendations for Future Operational Evaluation Efforts (Leidos SOW 2.2.3) 93
1.20. Operational Evaluation Framework. 93
1.20.1. Task 1: Installation and Configuration. 93
1.20.2. Task 2: Extract and Process Features. 93
1.20.3. Task 3: Run analytics, present results to analysts, assess findings. 94
1.20.4. Task 4: Collect, compute, and interpret metrics. 94
1.20.5. Proposed Operational Evaluation Schedule. 94

References 96

Appendix A – PRODIGAL Installation Guide 102
1.1. System Requirements. 102
1.2. Installation Guidelines. 103

Appendix B – PRODIGAL Introduction for Transition Partners 107
1.1. The Insider Threat Problem and PRODIGAL 107
1.2. Overview of the PRODIGAL Prototype System 107
1.3. Background: ADAMS program evaluations 108
1.3.1. Evaluation approach and metrics 110
1.3.2. Evaluation Results 111
1.4. Findings and future research 113

Introduction

This report highlights the key accomplishments and lessons learned by the Leidos PRODIGAL Team performing ADAMS tasks under contract W911NF-11-C-0088. The period of performance for this report is May 2011 through November 2016.

The report aligns with the the Research & Development elements from the PRODIGAL Statement of Work summarized below. Research in Phase 1 focused on devising state of the art algorithms for structural anomaly detection, integrating them into a system, and testing against live ADAMS program data *(commonly referred to in the report as "Vegas data")* with inserted red team targets. In Phase 2, while continuing to advance the fundamental detection algorithms, the focus of efforts shifted to explaining and combining algorithm output, supporting end-user analysis of potential insider threat activities, and preparing the system for transition. We will refer to this outline in subsequent sections.

1. Phase 1 Program Years 1 & 2
    - 1.1. Fundamental Research into Insider Threat Detection
        - 1.1.1. Research in scalable algorithms for network analysis (GTRI)
        - 1.1.2. Research in models of graph evolution (CMU)
        - 1.1.3. Research in ensemble methods for anomaly detection (OSU)
        - 1.1.4. Research in joint probability and causal reasoning models (UMass)
    - 1.2. Research Integration and Evaluation (Leidos)
        - 1.2.1. Counter-Intelligence Domain Analysis (Leidos)
        - 1.2.2. System Development, Integration, and Testing (Leidos)
    - 1.3. Surrogate and External (Program Data) Experimentation (Leidos)

2. Phase 2 Program Years 3, 4, & 5
    - 2.1. Research
        - 2.1.1. Research in massive temporal and visual dynamic graph analytics (GTRI)
        - 2.1.2. Research in graph summarization and understanding (CMU)
        - 2.1.3. Research in ensemble methods for anomaly detection and explanation (OSU)
        - 2.1.4. Research in causal dependence in complex domains (UMass)
    - 2.2. Research Integration and Experimentation (Leidos)
        - 2.2.1. End-User System Engineering (Leidos)
        - 2.2.2. Algorithm / Evidence Combination and Explanation (Leidos)
        - 2.2.3. Operational evaluation and transition preparation (Leidos)

Executive Summary

The Proactive Discovery of Insider Threats Using Graph Analysis and Learning (PRODIGAL) team, comprised of Leidos Advanced Solutions Group and our university partners, Georgia Tech Research Institute / Georgia Institute of Technology, Oregon State University, University of Massachusetts, and Carnegie Mellon University, lead a research and development program focused on anomaly detection and explanation applied to the insider threat domain. This paper reports on our fundamental anomaly detection and explanation research, and the development and experimentation with our prototype PRODIGAL insider threat detection system as a testbed for exploring a range of detection and analysis methods. The data and test environment, system components, and the core method of unsupervised detection of insider threat leads are presented to document this work and benefit others working in the insider threat domain. We also discuss a core set of experiments evaluating the prototype's ability to detect both known and unknown malicious insider behaviors. The experimental results show the ability to detect a large variety of insider threat scenario instances embedded in real data with no prior knowledge of what scenarios are present or when they occur.

Research activities of the PRODIGAL team under the ADAMS program fall into four major categories:

- Fundamental research into anomaly detection algorithms, models, and feature selection and normalization techniques;
- Domain-specific research to provide meaningful indicators and scenarios for detection and explanation of potential insider threat;
- Design and implementation of an experimental ensemble-based framework for research into insider threat detection, including all required engineering and data exploration over the ADAMS program database of user activities; and
- Experimentation over 16 months of program data seeded with dozens of inserted targets to compare, contrast, and combine detection methods.

Fundamental Anomaly Detection Research: Our university-based team members developed innovative algorithms for structural anomaly detection. Several employ statistical models and operate over populations of feature vectors in an unsupervised fashion to estimate the likelihood that a member was created by the same process that created the bulk of the population (e.g. RPAD, RIDE, RDE, Cross Prediction, Ensemble GMM). Significant effort was devoted to feature selection and normalization techniques, which have proven highly successful on the ADAMS program data. Temporal models were developed to find anomalous sequences of actions (Temporal-based Anomaly Detection, Vector Space Models). These approaches represent user activities as sequential patterns, possibly at multiple levels of granularity, and identify unlikely sequences. Additional research focused on leveraging the human analyst by providing methods for exploring huge graphs or high dimensional spaces for anomalous or unexplained behaviors (Oddball, Apolo). Finally, significant research efforts addressed ways of exploiting the graph structures inherent in large collections of human-computer transactions (STINGER, Community Detection, Seed Set Expansion).

Domain-Specific Insider Threat Detection Research: Based on prior research into insider threat, PRODIGAL team members developed a large set of features based on domain-specific

concepts such as activity type and their use in known threat scenarios. These include measures of computer user activity levels, statistical outlier measures of these activities, and ratios of activities designed to uncover unusual behavior. Features were implemented within the PRODIGAL framework for use in statistical outlier detection as well as by other structural anomaly detection algorithms. Complex pattern detectors for known threat scenarios were also implemented using a novel, graphical Anomaly Detection Language (Fraudster, Sabotage, IP Theft, Ambitious Leader).

Ensemble-Based Research Framework: Leidos developed an ensemble-based, unsupervised technique for detecting potential insider threat instances. The PRODIGAL framework ingests computer usage activity observations, computes features, operates over 100 anomaly detectors, and combines the results from the multiple detectors using an ensemble technique that provides the analyst with a single score for each user, for each day, serving as a starting point for further investigation as illustrated in **Figure 1.**

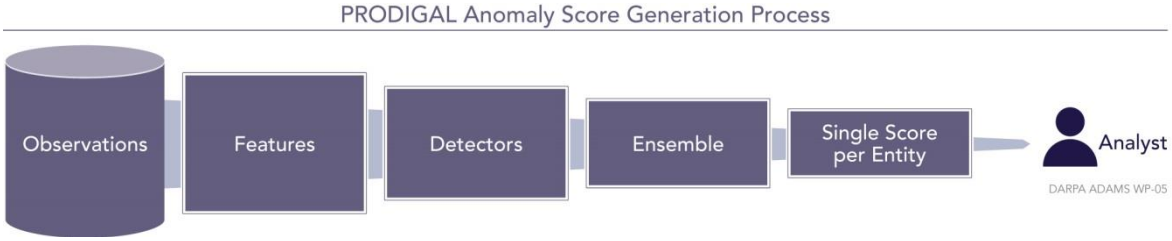


Figure 1 *Illustration of how PRODIGAL combines the results from multiple detectors with an innovative ensemble technique that provides the analyst a single anomaly score.*

We have also begun incorporating explanation capabilities with the ensemble approach so that underlying reasons for detection from individual detectors can be combined in the final result presented to analysts. These explanations are made available to the analysts in a user interface, with preliminary methods for incorporating user feedback within the PRODIGAL dashboard based upon the rich set of available features and anomaly detector scores. The report further describes the architecture of the prototype system, the environment in which we conducted these experiments and potential transition environments.

Insider Threat Detection Experimentation: When run over 16 months of real monitored computer usage activity augmented with independently developed and unknown but realistic, insider threat scenarios, this technique robustly achieves results within five percent of the best individual detectors identified after the fact. We discuss factors that contribute to the success of the ensemble method, such as the number and variety of unsupervised detectors and the use of prior knowledge encoded in detectors designed for specific activity patterns. This experimentation represents more than 1800 runs of detectors, varying by type of user activity, structural model (vector, sequence, graph), baseline population (peer group, shared resources, shared communications), as well as over time. This variety supports evaluation of robustness as well as sensitivity of detection methods.

Figure 2 illustrates how these activities have interacted and supported one another over the course of the ADAMS program.

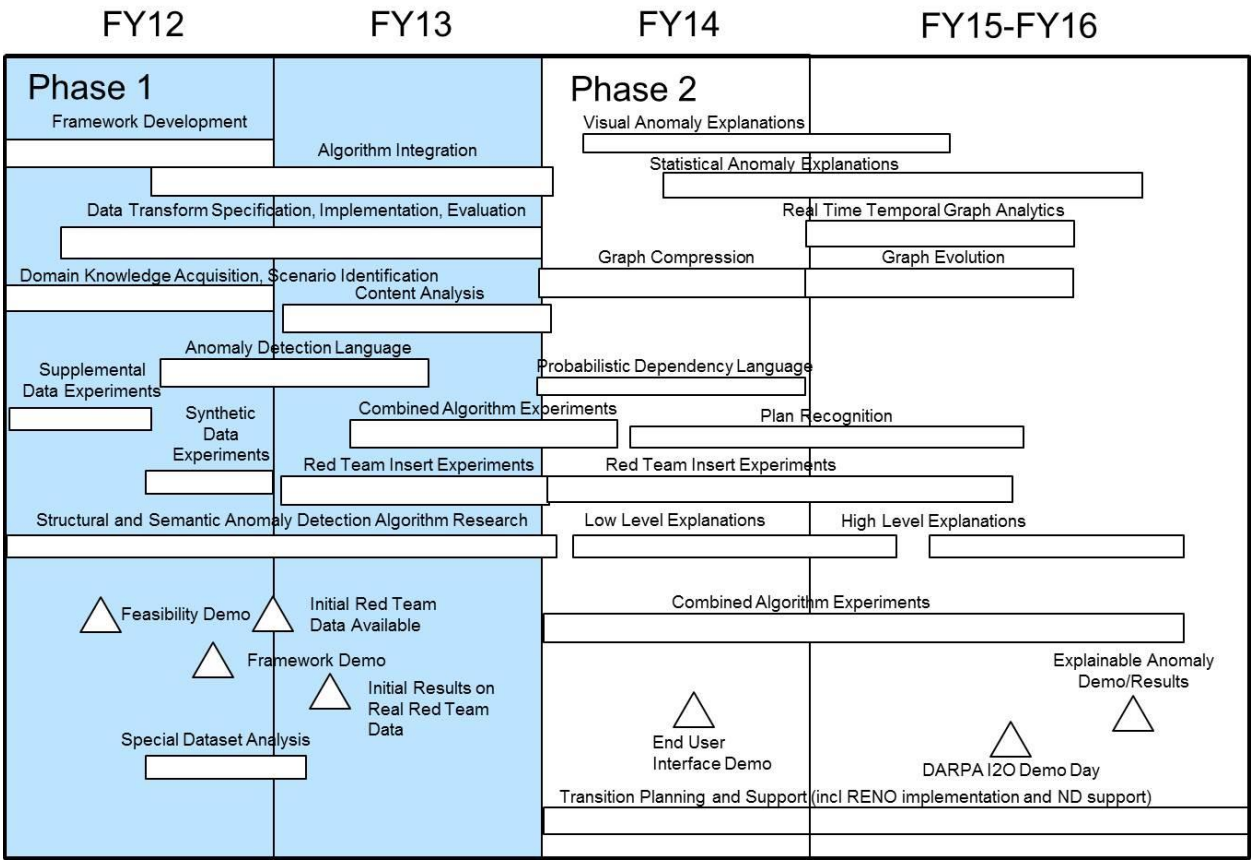


Figure 2 *How PRODIGAL activities have interacted and supported one another.*

Fundamental Research

1.1. Georgia Tech Research Institute (GTRI) Algorithm Research Activities

1.1.1. Research in scalable algorithms for network analysis (GTRI SOW 1.1.1)

GTRI-1: Massive-Scale Streaming Graph Algorithms

Under this task, Georgia Tech and GTRI developed and delivered an implementation of the Spatio-Temporal Interaction Networks and Graphs Extensible Representation (STINGER) dynamic temporal and semantic graph analysis data structure and toolset. During the course of this project, STINGER was expanded to support additional semantic capabilities, data ingest and exploration functionality, and exploration. STINGER and the algorithms developed on STINGER have been released as open source tools for the benefit of the graph analysis community and other organizations.

The core of STINGER was expanded to support multiple simultaneous analysis kernels running on the same massive graph set in shared memory space across processes. STINGER was also successfully integrated into the PRODIGAL framework and used to support the implementation of GTRI-2, GTRI-3, and GTRI-6. Analytics were implemented to aid in the use of STINGER to understand the results of other algorithms including breadth-first search, connected components tracking, methods of calculating local entropy, local-entropy based clustering, clustering coefficient tracking, k-core extraction, PageRank, and others. Support for additional properties and type tracking was added to STINGER. Participants in GTRI-1 were responsible for testing many of the other algorithms and running experiments on the Vegas dataset as well as performing a key portion of the integration task.

Query tools for the shared-memory graph structure were developed to enable local exploration of the relationships in the data set in a graph-focused manor. A web-based interface was developed to maintain control over the STINGER workflow and enable some level of visual exploration and analysis of the graph state. Using open web technologies also allows many analysts to use the system at the same time and gives users an interface that can be used on most platforms. Java interfaces were also developed to support additional algorithms and productivity interfaces. This included a working interface to support running GTRI-5 directly off of the graph data source.

Lastly, a large portion of this effort was spent examining the choice of graph structure and the mapping of events involving computers, files, people, emails, and more into graph substructures. Through mapping events into the graph in different ways, different community substructures and anomalous community transitions can be found.

Performance improvements to STINGER were published in [Ediger 2012]. However, need for massive scale analytics using this infrastructure did not materialize due to the limited size of the data. ADAMS opted for data to be limited in the number of users, but include as many events as possible (deeper and less broad).

GTRI-2: Massive-Scale Community Detection

GTRI-2 is a primarily a community detection algorithm that leverages fast safe parallel insertion/deletion in STINGER. Input graph is composed arbitrarily of email, printer, file, logon, url, domain, relationships as needed to produce different output communities. These can be used to determine what is typical behavior for a group to find anomalousness at finer granularity. The

detection is fully automated based on tight patterns of activity and resource sharing. The approach iteratively groups smaller community structures into larger communities to provide multiple resolutions. This exposes connections points between organizations and shifting allegiance.

Early in the GTRI-2 task, an approach and algorithm were developed to perform dynamic discovery and tracking of community structures within a graph. These were quickly implemented on STINGER and used with early datasets in addition to the later Vegas datasets. Through the course of the project, performance improvements to the algorithm were developed and the community detection was expanded to additional graph types including email, web, file, printing, logon, and combined graphs to provide groups and data to other parts of the PRODIGAL system for forming baseline comparisons. Different methods of scoring and optimizing community detection were explored and compared.

Once the community detection was delivered and integrated, work began on approaches to score and maintain the betweenness centrality of vertices in unweighted and undirected graphs without recomputing. The approach removes unnecessary and redundant work through maintaining additional data structures and is based on [Brandes 2001]. The algorithm follows the significance of all players in the graph and is the first algorithm that allows online analysis as computation time has been significantly reduced. It is easily parallelizable and can be applied to both exact and approximate computation. This metric was also integrated into the PRODIGAL system. The GTRI team performed full evaluation and development of the approach which was published in [Green 2012] and [Green 2013]. The latter work focused on the effect of data structure choice on the performance of the algorithm.

In order to better understand the evolution of community structure, the GTRI-2 team collaborated with CMU to examine graph generators. Specifically, the team set out to create generators that better mimic the development of the graphs of interest over time (as opposed to current generators which are designed to produce static graphs or graphs with a constant distribution).

Lastly this effort produced a new method for maintaining a labeling of the connected components of a dynamic graph without full recomputation even in the face of deletions. It is up to 128X faster than static computation in some cases, has low memory requirements so that it can be used on massive-scale graphs, and is a vast improvement over previous approaches that necessitated recompute.

The results of community detection were used in experiments on detector diversity (Section 1.9)

GTRI-3: Seed-Set Expansion at Multiple Scales

In the first few months of the project, GTRI-3 developed software that demonstrates a suite of graph algorithms for analyzing initial datasets and demonstrating immediate graph analysis capabilities within the PRODIGAL system. The algorithms included:

1. Graph Diameter – the longest geodesic (or “shortest”) path between any pair of vertices in the graph.
2. Clustering Coefficients – for each vertex, the probability that the neighboring vertices are connected. A coefficient of 0 means that the graph is locally tree-like; a coefficient of 1 means that the graph is locally a triangular mesh.

3. Connected Components – finds isolated subgraphs of the larger graph; the software computes the number of components, the number of vertices in the largest component, the average number of vertices per component, the average squared value of the number of vertices, the variance of the vertex number, and the standard deviation of the vertex number.
4. K-Betweenness Centrality – for vertex v, the number of paths between any other pair of vertices that pass through v and have length equal to the shortest path length + k, where k is a nonnegative integer.
5. Vertex Degree Distribution – the number of vertices with a specified number of edges.
6. Community Detection – uses the McCloskey-Bader algorithm to partition the graph into groups of vertices, with minimal connections between the groups.

Following this, GTRI-3 focused on seed set expansion for a streaming graph of batch updates and on detecting and executing the case of a community split without global recompute. This included looking at the effects of excluding low degree vertices from the initial community detection process. Examined datasets included the CERT data and the stack overflow datasets. Vegas data was included as it became available.

Novel methods for computing seed set expansions optimizing a number of metrics were theorized and implemented. Performance test were run on the Vegas dataset to compare and improve update methods. The approach was also extended to weighted graphs. Streaming seed set expansion can identify which vertices change communities frequently and/or have strong ties external to the community. It also allows a local subgraph of evidence and interaction to be extracted for deeper analysis following anomaly detection by another PRODIGAL algorithm.

Several de-agglomeration methods for updating communities after edge updates have implemented. Seed set expansion was also used to examine the area and interaction around known red team vertices to help understand anomaly reports and identify anomalous patterns. Although promising in support of analysis, this approach was not integrated into the PRODIGAL detection pipeline as a primary detector.

GTRI-4 Vector Space Models for Unsupervised Activity Classification

VSM breaks streams of discrete events into n-grams, and then compares the resulting feature vectors. In PRODIGAL, this is used to examine streams of logon logoff events, file access patterns, email frequency and more. The advance over previous incarnations of VSM is that we allow *unsupervised* anomaly detection. The algorithm can also process results produced by other algorithms to increase fidelity. The result is a score of the dissimilarity of the sequence to other sequences. On the September Vegas data, VSM ranked at least one actor in two scenarios very highly (top 5%) and four reasonably highly (top 30%). The lone actor in the final scenario was ranked at 41%.

Early in the project, the GTRI-4 team quickly transitioned to work with the RUU dataset, which was abstracted from the RUU format into a per-process stream of discrete events to be processed by the Vector Space Model algorithm. Each of these events was chosen to represent a pair of values from the raw data: (system action, result). The team explored additional representations beyond n-grams to enhance the capabilities of the VSM algorithm with respect to detecting anomalies that may not be locally apparent, and/or avoiding false positives for minor, benign event sequence reorderings. An example of such an improvement would be to produce multiple alternative event abstractions and to intelligently decide which alternative is most pertinent for a

given context. A literature search was performed to generate ideas for these enhancements. The team created refined a descriptive taxonomy of dataset types. This described both the CERT and RUU datasets using this formalism, as well as the input accepted by the VSM algorithm.

After this initial work on the RUU dataset, similar transformations and analysis were performed on the CERT dataset. Over two thirds of the malicious activity in the CERT set were detected by VSM. As the Vegas dataset became available it was also process. The VSM algorithm is completely integrated into the PRODIGAL framework. Members have also looked at the use of VSM in other flows/contexts and to extensions to the basic algorithm within PRODIGAL. Initial work was also performed on a clustering-based extension to VSM.

VSM was included in the PRODIGAL detector suite in both daily and monthly variants.

GTRI-5: Temporal-Based Anomaly Detection Exploiting Event-Sequences

Temporal-Based Anomaly Detection (T-BAD) uses a multi-scale model of user behavior and detector of co-occurring anomalous activities and indicators of cognitive stress. The team developed a multivariate algorithm to detect anomalous frequencies of specific cognitive behaviors. Such anomalies are indicators of nefarious activity. A Markov model to characterize normal/abnormal behavior is developed around the dataset and used for classification of events in a stream. Where VSM classifies as stream as unusual compared to other streams, T-BAD detects specific events within a single stream or set of combined streams. This also can be used to extend and enhance results produced by other algorithms. The system was fully successfully integrated into the PRODIGAL framework.

The team carefully tuned weights for logon, process anomaly scores and tested on all red team data. The algorithm is based around particle filtering techniques that improve results by more than three times on some datasets. It lends itself easily to heat map visualizations that can be quickly examined by analysts even over large datasets of users over longer periods of time. The stream processing can be tuned to detect anomalies at multiple scales with varying levels of resolution.

GTRI-5 provided personnel for onsite testing with the Vegas data. This included developing and integrating HMM/mixture models for process and keyboard data.

T-BAD was retired as not sufficiently accurate. The possible source of the problem was lack of accurate timing data; hence it was placed on hold pending improvements in the data infrastructure.

GTRI-6: Evidence Combination

The primary evidence combination approach is based on a Bayesian approach commonly known as belief propagation. This uses the relational information in STINGER to identify anomalies that are connected and are reported consistently by different detectors. Anomaly reports are projected onto the graph and used to suppress or reinforce local anomaly reports. This improves the rate and the rank of true positives by combining results of various detectors over STINGER. It also decreases false positives by weakening the anomaly scores for disconnected evidence.

GTRI-6 is fully integrated into STINGER and the PRODIGAL framework. Efforts included experimenting on the red team data within the CERT and Vegas datasets. Performance analysis of evidence combination results with respect to individual algorithm results and a naive approach

for combining anomaly results obtained up to 60% improvement in the evidence combined rankings in one of the algorithms.

Analysis of the performance aspects and parallelization opportunities within GTRI-6 was performed. GTRI-6 was tuned for fast performance and resolution so that new results can be added to the graph and resolved quickly as the data changes. Parameter tuning, optimization and sensitivity analysis experiments with evidence combination to monitor the quality of anomaly results on STINGER graph were also performed.

Initially a different approach was studied that was shown to be highly successfully on the early datasets. It used looking for correlations and statistical deviations across parameters in a traditional and more straightforward statistical approach. This approach was not as strong of a performer in later datasets. As such, it was abandoned in favor of belief propagation.

1.1.2. Research in massive temporal and visual dynamic graph analytics (GTRI SOW 2.1.1)

In June 2013, the ADAMS project transitioned from Phase I into Phase II. For Phase II GTRI was responsible for three tasks: 1) Dynamic Graphs, 2) Temporal Analytics, and 3) Learning Models for Anomaly Detection. Progress was steady during 2013. The GTRI team attended the ADAMS/SMIC PI meeting at DARPA Oct 2-4, 2013 presenting 4 posters and 1 partial oral presentation.

At the end of 2013, it was decided that the direction of the ADAMS project was no longer congruent with the GTRI tasks, and the project was discontinued. Below follows a short summary of the accomplishments of each of the three tasks.

Task 1: Dynamic Graphs

The goal of this task was to enhance the graph analytics to provide visualization of communities and influencers, interactive labeling and evidence combination, and online statistical analysis of graph kernels. The approach to this task was to identify anomalies as changes in group behavior or deviation in behavior from behavior of the group. This was accomplished by selecting a set of events by choosing feature types and temporal range, mapping events of each feature type to a set of edges and vertices representing the actors (user, PC, file, URL, etc.) and relationships involved, constructing the graph and applying an unguided community detection algorithm based on optimizing for dense communities that are sparsely connected to each other.

Results: The STINGER server was integrated into PRODIGAL via MySQL, a multi-process dynamic persistent STINGER server was implemented as an update and replacement for the existing server, and a fully generic data ingest tools for JSON, and CSV formats with XML were implemented. These extensions allowed for several actions. One, STINGER functionality was added to support arbitrary metadata and non-graph-based algorithms. Data replay from MongoDB resulted in more detailed statistical analysis over temporal extents of various resolutions. Two, algorithms can register to receive the computed results of other algorithms. Three, web visualization tools were developed with variable temporal range and batch resolution supporting: subgraph extraction and exploration, and additional PRODIGAL event details. Four, easy adjustment of graph ontology are possible.

Betweenness centrality was explored in two ways. One, research into better temporal approximations for betweenness centrality and possible visual extensions to community detection methods. This resulted in a new approach for root selection for approximate dynamic

between centrality being implemented, because improper root selection can lead to biased results. Two, parallel approximate betweenness centrality was implemented. A clustering coefficient algorithm was developed to support and combine optimizations of both implementations.

Task 2: Temporal Analytics: Interactive Graph Exploration & Feedback Incorporation

The goal of this task was to provide visualization of anomalous actions, and integration of statistical analysis. This task had three main approaches. One, perform statistical analysis of the temporal subgraph to allow identification of specific anomalous actions. Two, visualize connections among bad actors flagged by other algorithms. And three, aggregate "weak signals" using Belief Propagation to combine flags raised by algorithms into higher-confidence flags.

Subtask 2.1: Graphical context construction (Adaptive Navigation)

- Develop techniques that generate graphical context to help explain why an anomalous node is flagged (e.g., a small, relevant subgraph centered at the node). For example, a flagged user X may be a mastermind bridging five other suspicious users, and indirectly connected to another two.
- Develop technique to visualize all these aforementioned suspicious users and their connections. The construction may be automatic (e.g., include no more than 2-step-away neighbors) or user-driven (e.g., user interactively requests more nodes and edges be added).

Subtask 2.2: Interactive operations for manipulating graph visualization (GLO-STIX)

- Given a flagged anomalous node and its graphical context (say, generated by our construction techniques above), it is unlikely that a single visualization can fully explain the graphical context that helps the analysts understand the causes. The end user may need user-friendly, lightweight interactive tools to explore the graph and manipulate the visualization (e.g., node ordering, grouping, filtering, etc.).

- Develop graph-level operations (e.g., align or aggregate nodes by attributes, map visual elements like node sizes or colors to node attributes), which when combined and executed in sequence, can easily allow the user to transition from one familiar kind of graph visualization to another. Without these tools, analysts may need to user multiple, dedicated and independent visualization systems to perform similar analysts.

Results: Both subtasks described above have produced significant practical new ideas in visualization, human-computer interaction data mining and visualization. We have published our work and their related ideas in 8 papers (1 journal, 3 conferences, 2 posters and 2 demos). Our research has also formed the foundations of two Computer Science PhD students' theses (Mr. Robert Pienta, and Mr. Chad Stolper). Three undergraduate students have actively participated in this research and co-authored papers [Stolper 2014a] [Stolper 2014b] [Lin 2013] [Lin 2013a]. We disseminated our results via lectures in undergraduate and graduate classes at Georgia Tech and other universities, over 18 talks given nation wide and internationally at top venues (e.g., NIPS, CHI, SDM), and outreach talks for undergraduate students, and the general public.

The researchers in this task collaborated with ADAMS team members at CMU on temporal graph generation. The feasibility of web-based visualization engines was researched. The results of this research were then used to build a prototype interactive visualization for graphical explanation of anomalies. The Apolo framework was extended to work on various graphs stored

in MySQL. Research was conducted on graphical explanations for machine learning algorithms and graphical explanation techniques were formalized for several classes of machine learning algorithms. On-demand sense-making tools were developed to group and rank nodes by attributes.

Research contributions:

Subtask 2.1: Graphical context construction (Adaptive Navigation) [Pienta 2015a] [Pienta 2014] [Pienta 2015b] [Lin 2013] [Lin 2013a]

- We aim to reduce the visual complexity and clutter commonly created by large graphs. For example, typical scale-free networks have high-degree nodes that, when visualized, are connected to their hundreds or thousands of neighboring nodes. We developed a framework for locally exploring a graph without clutter, showing only the most subjectively most interesting nodes, and hence being *adaptive* to the users' interests (see **Figure 3**).
- We introduced a formal notion of subjective interestingness for graph exploration taking both divergence between local and global distributions, and similarity to explored nodes into account.
- A measure of surprise over neighborhoods – rather than local node attributes – to draw users in the direction of graph areas with subjectively interesting content.
- This subtask has formed the foundation for Computer Science PhD student Robert Pienta's thesis work on developing scalable visualization techniques for making sense of million and billion node graphs.

Figure 3. *(a) The Rotten Tomatoes movie graph shown using conventional spring layout (an edge connects two movie nodes if some users voted them as similar). Even for this relatively small graph of 17k nodes and 72k edges, a global visualization does not provide much insight. (b) A better way, using our FACETS approach, focuses on movies that are the most subjectively interesting, surprising, or both. For example, FACETS suggests Pretty Woman (romantic-comedy) as a interesting, surprising related movie of Miss Congeniality (crime-comedy).*

Subtask 2.2: Interactive operations for manipulating graph visualization (GLO-STIX) [Stolper 2014a] [Stolper 2014b]

- GLOs (Graph-Level Operations) is a new idea and model for specifying graph visualization techniques – they are like "LEGO blocks" for visualization. We provide these LEGO blocks; the user then flexibly combines them to create helpful visualization on demand. Example GLOs include: ranking a group of nodes by node attribute, grouping nodes into super nodes (see **Figure 4** below). We contribute a method to identify

GLOs and demonstrate the flexibility of GLOs in re-creating canonical graph visualization techniques.

- GLOs benefit researchers in helping them discover new network visualization techniques more easily and also benefit practitioners and developers in reducing engineering challenges.
- Graph-Level Operations is such a novel idea that it has become Computer Science PhD student Chad Stolper's thesis (co-advisors Prof. John Stasko and Polo Chau).

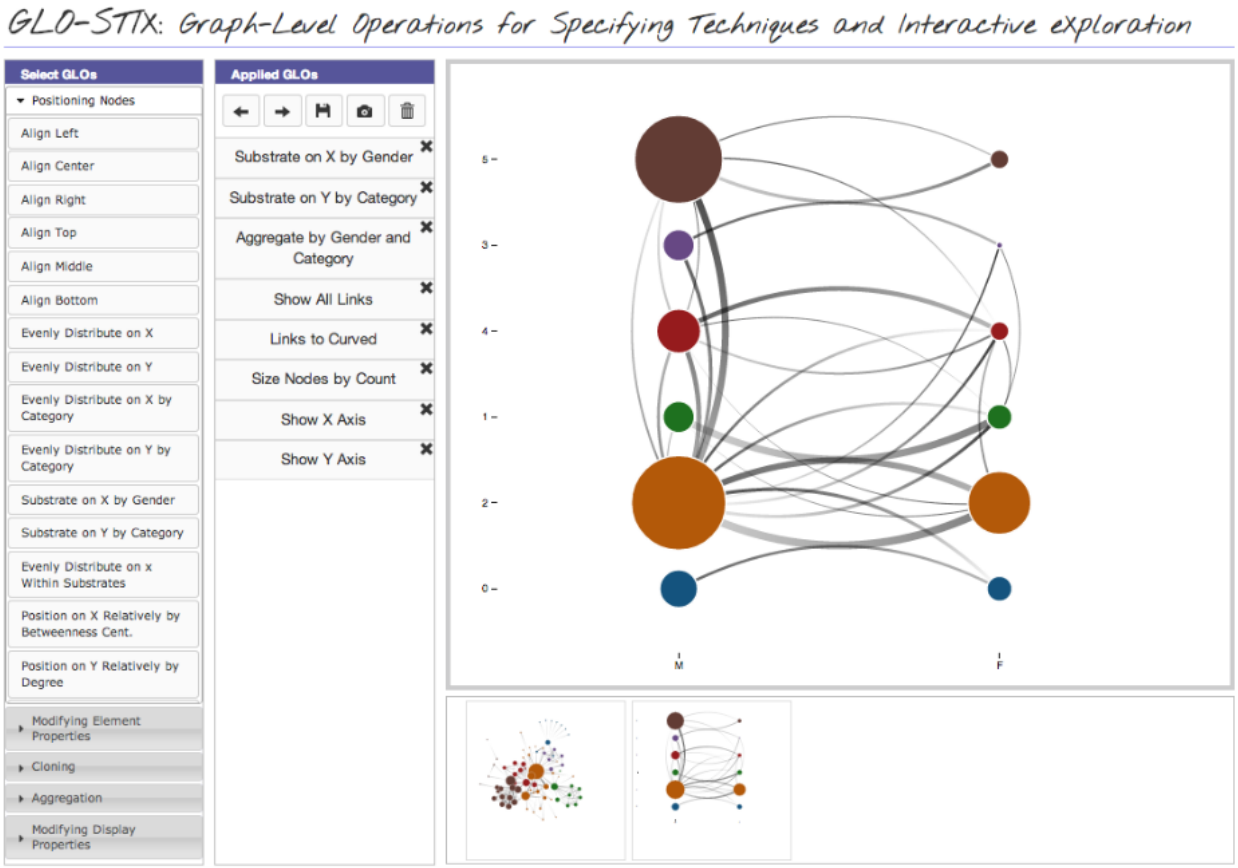


Figure 4. *A screenshot of the GLO-STIX user interface showing a user exploring the Les Miserables character co-occurrence graph using graph-level operations (GLOs). Nodes are characters, and an edge connects two characters if they co-occur in a chapter. The original node-link view of the graph is saved by the user as a snapshot in the bottom pane. From the list of operations available (shown in left-most column), applying those selected in the middle column transforms the original graph into the PivotGraph visualization displayed in the main view.*

Task 3: Learning Models for Anomaly Detection

The goal of this task was to implement, enhance, and integrate Optics-OF and SoNG and identify semantic interpretation of anomalies.

- Develop a newer set of advanced methods for anomaly detection that leverage an iterative bottom-up and top-down approach to support automatic explanations of anomalies will be advanced. A mixed-initiative approach and interactive exploration of anomalies will be developed to support PRODIGAL in analyses where explanations do not have clear conclusions.

- Operate on a "sack-of-n-grams'' (SoNG) representation. This preserves the local structural information in the event stream, and narrows the conceptual gap between BoW and sequential models. Experiments have shown that capturing some of the local temporal structure of events using n-grams does allow for the detection of inserted synthetic anomalies in a large corpus of collected data in many cases with reasonable confidence. More advanced algorithms for anomaly detection, applied on top of the SoNG representation, may allow for substantial improvement in detection accuracy. In particular, the application of density- based clustering methods (such as OPTICS) for outlier detection in insider threat scenarios may prove effective. These techniques can be combined with n-gram based feature vectors to effectively identify anomalous user behavior based on deviations in the local temporal structure of the event sequences they generate. That is, continue to capture local temporal structure in the event streams by using the SoNG approach for featurization of the streams. However, we propose the implementation of an outlier detection mechanism over these featurized streams based on OPTICS-style density based clustering.

This task resulted in 3 publications including a best workshop paper [Sharma 2015] and formed part of the theses of two PhD students. This research explored several ideas based on activity modeling to find patterns and anomalies, such as applying SMO-KML (Sequential Minimal Optimization - Multiple Kernel Learning) to learn both the kernel and SVM parameters, and to aid in clustering and classifying activities [Sharma 2014], addressing the limitations of OPTICS frameworks. We also explored the applying concept of Sequential Motion Textures (SMT) to model repetitious activities, by leveraging the representation of movements to model dynamic information.

Optics-OF. Optics-OF is an approach to anomaly detection within the framework of density-based clustering. To learn more about the details of the algorithm, please see [Breunig, 1999]. For our purposes, the benefits of using this algorithm as opposed to the previous, more ad hoc method that had been layered on top of the VSM cosine similarity computation are twofold.

First, OPTICS-OF allows us to move away from a more ad hoc approach that involves a "magic number" parameter and which makes assumptions about the distribution of the input data that may not hold. The ad hoc anomaly detection method layered over VSM would look at the average cosine similarity with the nearest k (usually, we chose k=10) neighbors, and used this value as the anomaly score. While this approach worked well in some settings, it is sensitive to scale and may make decisions about anomalousness that are intuitively quite odd if the density of the input distribution is far from uniform. Given that our data (variable-length vectors of discrete events representing user activity at workstations) is likely to have a non-uniform density, due in part to latent factors such as variable user roles within the larger organization, the ad hoc approach is likely to be insufficient for some subproblems (problem framings) that arise within the context of the ADAMS project. Outlier/anomaly detection built on density-based clustering provides a more principled approach, where a data point is considered anomalous if it is more distant from its neighbors than its neighbors are from each other – and thus it should address the issues of the ad hoc approach. Further, OPTICS(-OF) is more flexible than some other density-based clustering approaches, in that it computes results that effectively test a range of parameter settings at once, thus decreasing the dependence on proper selection of "magic number" parameters.

Second, OPTICS-OF can serve as a springboard to more interactive, "human-in-the-loop" exploration and visualization of datasets, with a specific eye towards identifying anomalies. Again, this is because of the more limited reliance of OPTICS on specific parameter settings. The data gathered during a single run of the algorithm is sufficient to allow the user to browse the clusters that are formed at multiple scales (as in hierarchical clustering), effectively visualizing clustering results with multiple parameter settings. It is our sense that this kind of interactive data exploration may be useful as the ADAMS/PRODIGAL project moves towards a deployed system, in terms of allowing the human to guide processing (select problem framings to refine results).

One notable extension to OPTICS-OF was needed for this setting. Specifically, the algorithm as designed does not deal well with duplicate points (in our case, exact duplicate event vectors). The authors suggest a method for dealing with the issue (only examining *unique* neighbors), but this method is not acceptable for our problem setting, because it will result in high anomaly scores for event vectors that are frequently duplicated but for which there are few "near misses". Our solution is to add a small amount of "background radiation" – a very small positive value – to a part of the outlier measure computation. This avoids a divide-by-zero problem without impacting relative score computation.

Initial Results. At this point, only a very preliminary set of results have been obtained and analyzed. A recent overhaul moved VSM from working at a scale of months to a scale of days, and it has not yet been possible to fully analyze these results and tune the algorithm appropriately (e.g. extend and adjust inputs as necessary for the altered time scale, see why "anomalous" users may have had anomalies detected more quickly on seemingly uninvolved days, etc.). Results for VSM-Ad Hoc (lift and ROC) are shown in **Figure 5** and **Figure 6**.

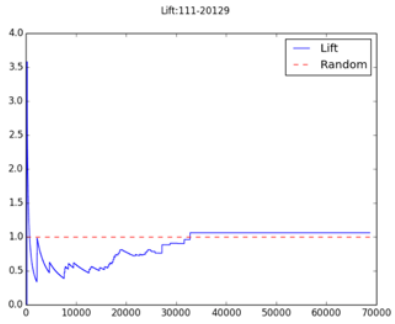


Figure 5. *VSM-Ad Hoc lift*

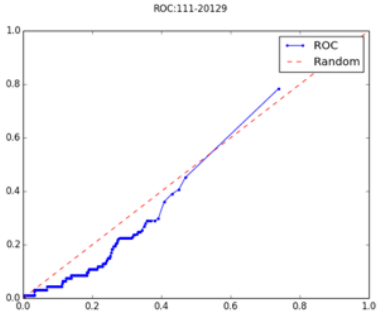


Figure 6. *VSM-Ad Hoc ROC*

As can be seen, although when operating on a scale of months (scoring each user based on their most anomalous day within a month) VSM did reasonably well, it is not outperforming a random baseline in this new configuration. More analysis needs to be done to understand why this is the case, and make appropriate adjustments for this new configuration. Results for VSM-OPTICS-OF are shown in **Figure 7** and **Figure 8**.

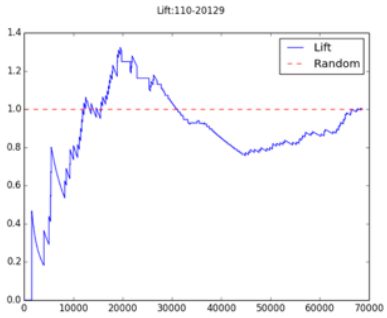


Figure 7. *VSM-OPTICS-OF lift*

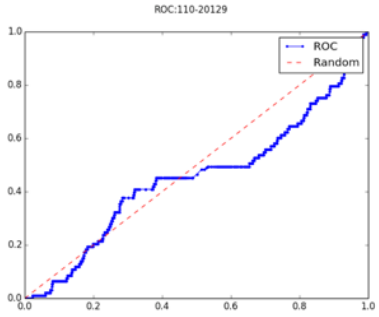


Figure 8. *VSM-OPTICS-OF ROC*

Again, these results, while possibly slightly better, do not appear to be significantly better than a random baseline. Because VSM-OPTICS-OF has only been run in the user-day configuration, it is not clear how its performance would compare to that of VSM-Ad Hoc in the month-based configuration in which VSM has been able to exceed baselines. In both cases, more analysis and tuning will be required to understand and improve results.

Running OPTICS-OF Over VSM. The current implementation of OPTICS-OF depends upon the previous execution of the main VSM code, in which an output file containing cosine similarity differences between vectors in the input dataset is created. This file is then used in the OPTICS-OF computations. Thus, the current procedure consists of first running VSM-Ad Hoc for the data in question, using the PRODIGAL framework as usual. This code has been baselined, and is described in detail in the baseline document. Then, OPTICS_OF_VSM.java should be adjusted to reflect the desired input/output directories and files. All relevant parameters can currently be found at the top of this java source file. This code can then be run to compute the OPTICS-OF anomaly scores for each user-day. Finally, OPTICSExpandData.java can be run to compute secondary metrics from these raw scores. Also included in the current codebase are OPTICS_VSM.java, which performs the basic density-based clustering (rather than computing anomaly scores), and a prototype GUI that enables visualization of the clusters, roughly in the manner described in [Ankerst 1999].

1.2. Carnegie Mellon University Algorithm Research Activities

1.2.1. Research in models of graph evolution (CMU SOW 1.1.2)

CMU-1: Oddball

Oddball has been shown to work on large weighted graphs, without human intervention. The main idea behind Oddball is to summarize each node's neighborhood subgraph, called the node's "egonet" (which include all nodes within one step away, and all edges in the neighborhood), using a set of features.

Oddball has been used in identifying the core set of features that successfully flagged anomalous nodes (and discovered patterns), such as the egonet's total edge weight, principal eigenvalue, and total number of edge. It has also been used to devise unsupervised methods that automatically correlate pairs of features (e.g., edge weight vs. principal eigenvalue) and pinpoint nodes whose features deviate from the rest.

In Phase 1, we delivered Oddball in Matlab to automatically flag anomalous/malicious nodes and have extended this effort of automatic anomaly detection in CMU-5: G-FADD.

CMU-2, 4: Interactive local graph exploration and visualization (Apolo)

Anomaly detection algorithms often only identify nodes of interest, providing little justifications. This implies analysts will likely need to reason or make sense of such detection results. In other words, they will need to engage in rapid, "bottom-up" sensemaking investigation to find supporting evidence, such as other potential nodes in the graph that may have attributed to the anomaly.

Apolo supports this "bottom-up" graph mining task by using a mixed-initiative approach---combining machine learning, visualization, and rich user interaction---to guide the user to incrementally and interactively explore and make sense of very large graphs. Apolo engages the user in an inductive, bottom-up approach, where the user gradually builds up an understanding over time by starting small, rather than starting big and drilling down. The user specifies some nodes of interest in the graph and Apolo's machine learning method called Belief Propagation learn from these exemplars to infer which other nodes may also be of interest.

During the course of the project, we have extended this interactive local graph exploration and visualization tool to incorporate analysts' relevant feedback and shipped Apolo package in Java. The ADAMS user will bring the source article into the interface, which initiates Apolo's Belief Propagation (BP) machine learning algorithm to find 10 most relevant articles and add them to the interface (so as not to overwhelm the user with too many articles). The user can categorize any number of the articles into an arbitrary number of user-specified groups (e.g., Information Visualization in blue, Collaborative Search in orange, Personal Information Management in green). Adding an article into a group makes it an exemplar for that group and causes the BP algorithm to infer the relevance of all other nodes in the network to the exemplar(s). If the user agrees with the algorithm's inference, he can ask Apolo to suggest more relevant articles to deepen his understanding. Sometimes, BP does not label articles as intended. The user can correct this by manually putting them into to the groups that he sees fit (making them exemplars). Apolo will support the ADAMS user in iteratively refining the groups; the user can split a group into finer groups, or merge multiple ones, and move articles between them.

CMU-3: Peta-scale graph mining system (PEGASUS)

Some of the ADAMS input data, hence the resultant graphs, may eventually become too large to fit in the main memory. As a result, many important algorithms that assume the graph would fit in memory will fail to run.

During the course of the project, we conducted forefront research in reformulating and developing such algorithms (e.g., finding connected components, computing graph radius) so that they work on huge graphs by developing massively-scalable graph mining algorithms by extending the algorithm development of Pegasus, an open-source, petabyte-scale graph mining system. Pegasus runs in a parallel, distributed manner on top of Hadoop, an open-source cloud-computing platform that implements the MapReduce framework.

As a result, the PEGASUS package we shipped in Java can extract following from the given graph data:

- PageRank: Measures the relative importance of nodes via the nodes that link to it.
- Random Walk with Restarts (RWR): Defines a relevance score between two nodes in a graph.
- Connected Components: Partitions the graph into groups of nodes mutually connected via their edges.
- Diameter: Measures the distance of the furthest-apart nodes in the graph.
- Degree distribution: The number of in- and out- edges among nodes of the graph.

We also have produced research papers with the details and implemented several new, important algorithms on top of the Pegasus framework that can be used in a wide variety of anomaly detection algorithms,

- Eigensolver: Finds the eigenvalue decomposition of the graph's adjacency matrix. [Kang 2011]
- BP/FastBP: An efficient way to solve inference problems based on local message passing. [Koutra 2013b],

CMU-5: Grid-based Fast Anomaly Detection Given Duplicates (G-FADD)

Many traditional outlier detection methods are slow due to the big number of duplicate points that the outlier detection literature has ignored before. Given a cloud of multi-dimensional points, GFADD detect outliers in a scalable way by taking care of the major problem of duplicate points. Fast Anomaly Detection given Duplicates (FADD) solves duplicate problems by treating them as one super node rather than considering them separately. Moreover, Grid-based FADD (GFADD) applies a k-dimensional grid on the k-dimensional cloud of points, and treats as super nodes only the grid cells that consist of more points than the number of nearest neighbors we are interested in. This method achieves near-linear runtime given duplicates, while Local Outlier Factor (LOF), the traditional outlier method that consists our baseline, has quadratic runtime. GFADD can spot anomalies in data sets with more than 10M data points, while the traditional LOF algorithm runs out of memory even for 20K data points. (See **Figure 9**)

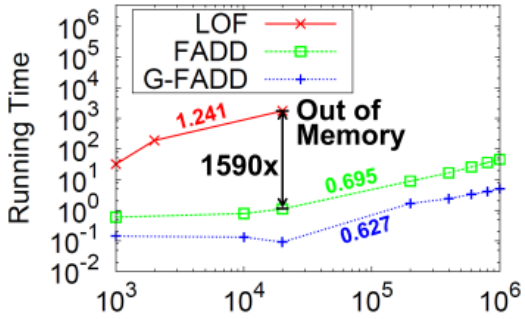


Figure 9. *FADD and GFADD vs LOF*

During the course of this project, Carnegie Mellon University developed and delivered an implementation of the *G-FADD* and experimented with PRODIGALNET dataset. Additionally with the work on G-FADD, a paper was published on CMU SCS Technical Report 2012 and a poster was presented on WWW 2013.

Thirteen variants of this algorithm have been implemented in the standard PRODIGAL suite of detectors. Each variant considers a pair of features found to be useful in detecting target insiders.

1.2.2. Research in graph summarization and understanding (CMU SOW 2.1.2)

The main idea is that compression and understanding go hand-in-hand: If a graph is, eg., a full clique with all N nodes connected to all others, we can easily describe/compress it, and thus understand its structure.

Most of the proposed subtasks are exactly based on this idea, that compression is understanding, and in-ability to compress a node or subgraph, is a sign of an anomaly.

Subtask 1 - STATIC COMPRESSION: The goal is to automatically extract the few most important parts of the graph. For example, a few 'stars' with huge degrees; and a few near-cliques of 10-15 nodes (conspirators?) Our approach is to use entropy/compression arguments: a subgraph (eg., star) is worth mentioning, if it helps us compress (=understand) the large graph.

Subtask 2 - VISUALIZATION: Following up, propose to allow visualization of a graph and its important subgraphs. We propose to study (and eventually, automatically choose) between two choices: (a) a spring-force layout of a (sub)-graph, which is suitable when a graph is small and (b) a careful 'spy-plot' of the adjacency matrix, which is more suitable for larger graphs.

Subtask 1: Static Compression

How can we succinctly describe a million-node graph? Given a graph with millions and billions of nodes and edges, how can we find its most “important” structures in a few sentences, and visualize its most important aspects? How can we measure the “importance” of a structure? These are exactly the problems we focus on.

Our main ideas are to create a “vocabulary” of subgraph- types that we know often occur in real graphs (such as stars, cliques, chains) shown in **Figure 10** below, and use this vocabulary to succinctly describe the large graph. We propose to measure success in a principled way, by means of MDL (Minimum Description Length). As such, a structure (= subgraph) is important, if it helps decrease the description length.

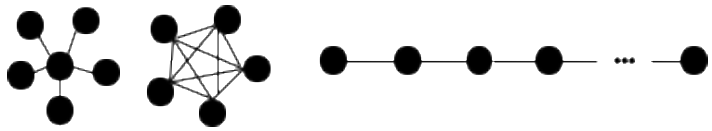


Figure 10. *Examples of Star, Clique, and Chain graphs addressed through static compression.*

CMU contributions are three-fold: (a) formulation: we provide a principled (MDL-based) encoding scheme to choose which vocabulary sub-graphs to use; (b) algorithm: we propose VoG, an efficient and effective method to minimize the description cost and (c) applicability: we report experimental results on numerous multi-million-edge real graphs.

Figure 11 shows that VoG is an efficient method.

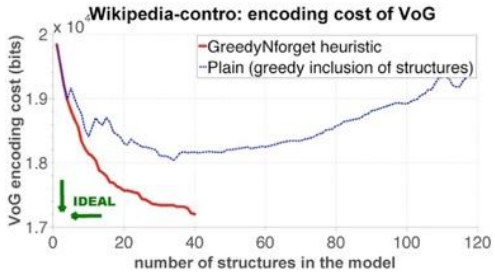
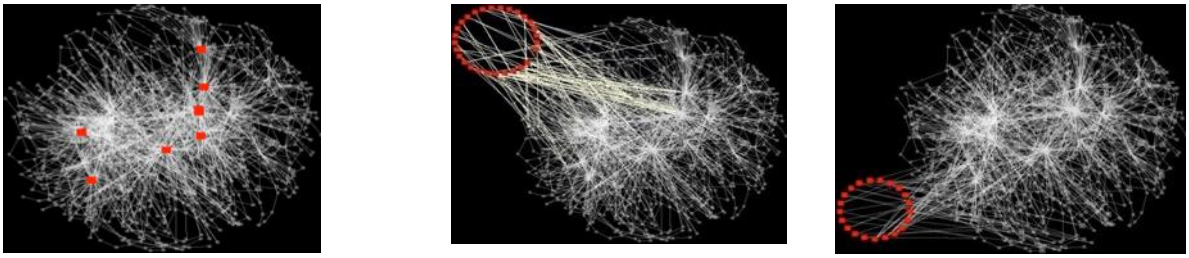


Figure 11. *Efficiency of VoG*

Figure 2 shows effectiveness of VoG through experimental result on Wikipedia graph.



top-8 star structures: admins, heavy wiki users, bots

top-1 and top-2 bipartite cores: edit wars. ***Left****: warring factions changing each-other's edits.* ***Right****: between vandals*

Figure 2. *Effectiveness of VoG through experimental result on Wikipedia.*

Subtask 2: Net-Ray

How can we visualize billion-scale graphs? How to spot outliers in such graphs quickly? Visualizing graphs is the most direct way of understanding them; however, billion-scale graphs are very difficult to visualize since the amount of information overflows the resolution of a typical screen.

CMU proposes NET-RAY, an open-source package for visualization- based mining on billion-

scale graphs. NET-RAY visualizes graphs using the spy plot (adjacency matrix patterns), distribution plot, and correlation plot which involve careful node ordering and scaling. In addition, NET-RAY efficiently summarizes scatter clusters of graphs in a way that finds outliers automatically, and makes it easy to interpret them visually. To the best of our knowledge, NET-RAY is the first work for visual mining on billion-scale graphs.

Extensive experiments show that NET-RAY handles very large graphs with billions of nodes and edges efficiently and effectively. Specifically, among the various datasets that we study, we visualize in multiple ways the YahooWeb graph which spans 1.4 billion webpages and 6.6 billion links, and the Twitter who- follows-whom graph, which consists of 62.5 million users and 1.8 billion edges. We report interesting clusters and outliers spotted and summarized by NET-RAY.

Figure 3 shows the experimental results of Net-Ray on Twitter and Yahooweb. The red circles are the outliers that Net-Ray has captured automatically.

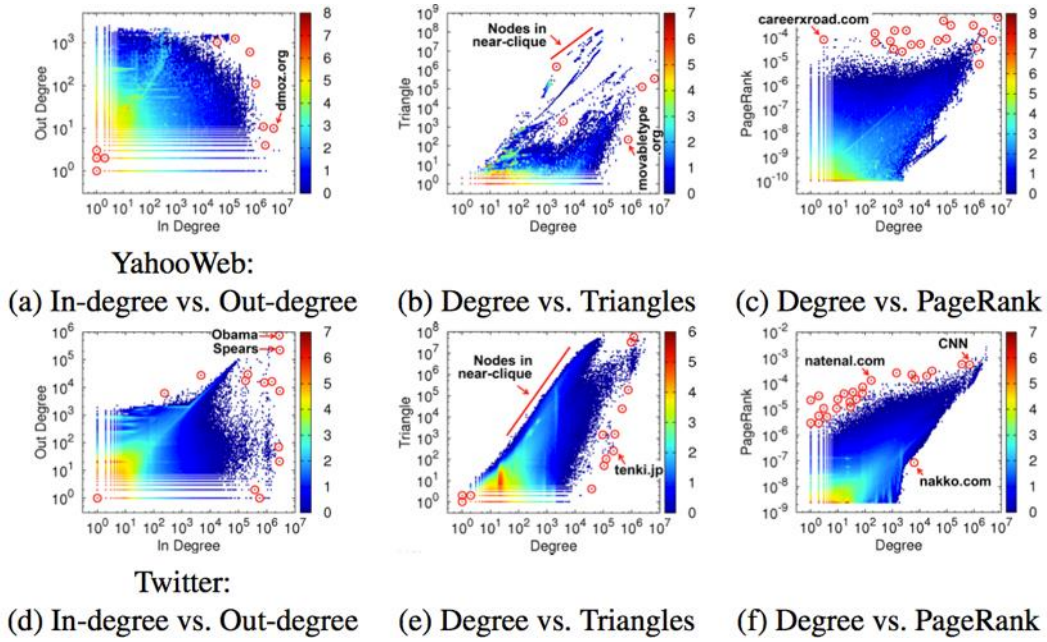


Figure 3. *Experimental results on YahooWeb and Twitter. Outliers in correlation plots give rich information on the structural patterns of graphs.*

1.3.Oregon State University Algorithm Research Activities

1.3.1. Research in for anomaly detection (OSU SOW 1.1.3)

During Phase I Oregon State University worked on the development and evaluation of novel anomaly detection algorithm as well as creating a novel framework for evaluating anomaly detection algorithms. Below work on the three most successful anomaly detection algorithms will be described, followed by a description of the new evaluation framework.

OSU-2: Cross Prediction

The main idea of the cross prediction method is to learn models to predict the value of each feature in terms of other features. Given a new instance x an anomaly score is then provided based on how well each of the features can be predicted in terms of the others. The assumption is that anomalous data will have certain attributes that do not exhibit the same conditional distribution characteristics as those attributes in normal data.

More specifically given a set of data, most of which is assumed to be normal, a conditional probability model is learned for each feature x_i in terms of the remaining features denoted by x_{-i}. We denote this probability model as $p(x_i|x_{-i})$. Given a new instance with vector x the anomaly score given to the instance is $P(x) = 1 - \prod_i p(x_i|x_{-i})$. This assigns a high score to instances where certain attributes are not well-predicted in terms of the others.

A desirable feature of this approach is that it has the potential to automatically scale for systematic trends in the data. For example, some users have many more logins than other users and the corresponding statistics of other attributes will be correspondingly scaled. The cross prediction approach has the potential to learn the general correspondence between the number of logins and scales of other attributes. In contrast more traditional anomaly detections techniques would need to learn components that characterize the full range of scales.

A potential weakness of the above approach is that the anomaly score can potentially be dominated by the scores of features that are inherently unpredictable. To address this issue a contribution of our approach is to develop a feature selection technique where we only consider features in the model that are predictable above a certain accuracy level. This improves performance substantially and also reduces the computational complexity of applying the model. Our current implementation uses regression trees as the base predictors, which are focused on predicting the quantiles of each attribute.

Overall the performance of this approach is not as high as the approaches described below on the insider threat data. The precise reason for this is not clear and is a topic of current investigation.

OSU-3: Ensembles of Gaussian Mixture Models (EGMM)

Gaussian Mixture Models (GMMs) are a commonly-used parametric model for density estimation of vector-valued data. As such, a natural approach to anomaly detection would be to fit a GMM on a dataset and then rank instances according to the negative log-likelihood assigned by the GMM. Anomalous instances will generally be expected to have smaller log-likelihoods and hence will appear higher in such a ranking. Unfortunately a single GMM is not a very robust density estimator due to the non-convex form of the associated parameter optimization problem, which we solve via the Expectation-Maximization (EM) algorithm. Thus, the anomaly rankings provided by a single GMM will often not be very robust across multiple runs of EM. Further, the number of Gaussian components *k* in the model is a difficult parameter to set and the anomaly scores can depend very much on the precise value.

To improve robustness, we generate a diverse set of models with multiple clustering assumptions about the value of *k* and random EM initializations and then combine those models to obtain the final anomaly score. Specifically, in our Ensemble GMM (EGMM) approach, instead of using a single value of *k*, we generate an ensemble of GMMs for a set of *k* values (denoted *K*). This is easier to specify and allows the algorithm to perform model selection automatically. For each value of *k* in *K* we use *r* runs of EM with random initializations to produce a set of *r* GMM

models. In addition, we use boostrap aggregation (bagging) to help ensure that models for an individual *k* are independent of one another. Ideally, all models should be independent if we expect to improve generalization by combining their results.

Given the produced ensemble of GMMs and a data instance *x* we assign a likelihood to *x* by averaging the likelihoods assigned to across the ensemble. Note that this average score can be interpreted as a valid PDF since the average of a set of PDFs is also a PDF. Note that this approach is highly parallelizable if desired since the training of individual GMMs is independent.

More recently we have observed that an alternative model combination approach performs better on some benchmark datasets. In particular, averaging the log-likelihoods of the individual models improved performance over just averaging likelihoods. This combination method has an information theoretic interpretation as computing the average surprise (negative log likelihood) of each data point.

OSU-8: Grouped Ensembles of Gaussian Mixture Models (GEGMM)

The GMM and EGMM methods can be viewed as organizing data points into a collection of clusters. These algorithms treat every data point as independent. In the context of insider threat detection, each data point represents one user-day. The user-days corresponding to a single user should not be treated as independent. For example, consider a clerical worker whose computer is compromised so that the worker appears to start behaving like a system administrator as of March 1. A method such as EGMM that treats each day independently will cluster the days prior to March 1 with days from other clerical workers and cluster days after March 1 with system administrators. It will not detect the compromise. The Grouped-EGMM method extends EGMM to require that all user-days from the same user must be assigned to the same cluster. The data from the compromised clerical worker will not fit well in any cluster, so it will be detected as an outlier.

We compared GEGMM to EGMM on data from Vegas, September 2012. **Figure 4** shows that the AUC is substantially improved.

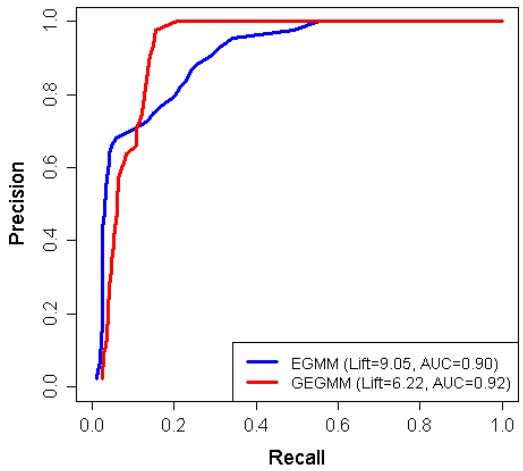


Figure 4. *Comparison of EGMM and Grouped EGMM on Vegas, September 2012*

OSU-4: Repeated Impossible Discrimination Ensemble (RIDE)

In machine learning, we often fit highly flexible models to training data. Such models can easily "overfit" the data, which means that the model discovers ad hoc patterns that do not generalize to new data and therefore lead to errors during prediction. As a general rule, outlier data points are easier to overfit than data points that belong to a tight cluster. The idea underlying the RIDE algorithm is to exploit overfitting as follows. Given a set S of data points (vectors of features), randomly partition it into two equal-sized sets S_1 and S_2. Label all points in S_1 as belonging to class "0" and all points in S_2 as belonging to class "1". Then apply a flexible learning algorithm, such as boosted regression trees, to learn a classifier that can discriminate the "0" data points from the "1" data points. This is an impossible discrimination task because, by construction, the two sets S_1 and S_2 have exactly the same probability distribution. A properly-calibrated classifier would therefore predict that the probability of any data point x belonging to class "0", denoted $P(y = \text{"0"}|x)$, should be the same as the probability of belonging to class "1", denoted $P(y = \text{"1"}|x)$, and should be 0.5. However, if we allow the classifier to overfit the data, it will discover ad hoc reasons for why the data points in S_1 are from class "0" and the data points in S_2 are from class "1". We can use the $|P(y = 1|x) - 0.5|$ as a measure of how badly the classifier has overfit data point x.

The Repeated Impossible Discrimination Ensemble (RIDE) method performs this random-split-and-overfit process many times. For each data point x, it computes the average value of $|P(y = 1|x) - 0.5|$ and uses this as an anomaly score. We call this general paradigm "Anomaly Detection by Overfitting" to contrast it with standard approaches, which can be viewed as "Anomaly Detection by Underfitting" because they fit a probability model and then score as anomalies those points that do not fit the model. An advantage of the RIDE approach is that it can work well when there are irrelevant features, because it can apply feature selection methods from supervised learning.

OSU-5: Isolation Forest (IFOR)

Our benchmarking study (see below) determined that the Isolation Forest algorithm of Liu, Ting & Zhou [Liu 2008] was an excellent anomaly detection algorithm, so we added it to our collection of anomaly detectors. An *isolation tree* is a randomized tree that recursively partitions the data space into axis-parallel rectangles. In each recursive step, the algorithm splits a node by selecting one dimension of the data at random, computes the minimum *min* and maximum value *max* of that dimension for the data points that have reached the current node, and selects a threshold uniformly at random between *min* and *max*. The data points are then sent to two child nodes according to this threshold. Splitting stops when there is exactly one data point reaching a node. The node is defined as a leaf node, and the data point is said to be *isolated* in that leaf. To compute the anomaly score for a query point, the point is dropped through the tree until it reaches a leaf, and the depth d of the leaf is computed. The intuition is that points having low isolation depth are anomaly points that are easily separated at random from the normal data points. An *isolation forest* is an ensemble of 100 isolation trees. Each tree is built using only a random subsample of the full data set. Our experiments suggested that the optimal subsample size should be around 2000 (whereas Liu et al. suggest 256). To compute the overall anomaly score for a query point, the average isolation depth $\bar{d}$ of the point is computed across all of the trees in the forest. The anomaly score is computed as $2^{-\frac{\bar{d}}{\mu}}$, where μ is a theoretically-computed expected isolation depth.

A Framework for Benchmarking Anomaly Detection Algorithms

Research in anomaly detection suffers from a lack of realistic and publicly-available problem sets. This makes it difficult to draw general conclusions about the relative effectiveness of different anomaly detection methods. With this motivation, we developed a framework for constructing anomaly detection benchmarks from real-world classification data sets.

We started by identifying key properties that anomaly detection benchmarks should possess as follows:

Requirement 1: Normal data points should be drawn from a real-world generating process. Generating data sets from some assumed probability distribution (e.g., a multivariate Gaussian) risks not capturing any real-world processes. Instead, as the field has learned from many years of experience with benchmark problems, it is important that the problems reflect the idiosyncrasies of real domains.

Requirement 2: The anomalous data points should also be from a real-world process that is semantically distinct from the process generating the normal points. The anomalous points should not just be points in the tails of the normal distribution.

Requirement 3: Many benchmark datasets are needed. If we employ only a small number of data sets, we risk developing algorithms that only work on those problems. Hence, we need a large (and continually expanding) set of benchmark data sets to ensure generality and prevent over fitting.

Requirement 4: Benchmark datasets should be characterized in terms of well-defined and meaningful problem dimensions that can be systematically varied. An important goal for benchmarking is to gain insight into the strengths and weaknesses of the various algorithms. Ideally, we should identify those dimensions along which anomaly detection problems might vary and then generate benchmark data sets that vary these dimensions in a controlled fashion.

There is currently no established set of problem dimensions for anomaly detection and we expect this set to evolve with experience. However, in this initial work we proposed a set of four dimensions: (a) point difficulty, (b) relative frequency, (c) clusteredness, and (d) feature relevance/irrelevance. The full paper on this research [Emmott et. al., 2013] defines these properties more formally. In addition it specifies an algorithmic methodology for generating anomaly detection datasets from real-world classification and regression data that can span the range of these properties.

The result of our initial work is a set of thousands of anomaly detection problems derived from real-world data that span the range of the above properties. Our work in [Emmott et. al., 2013] also describes an empirical study where we apply a number of state-of-the-art anomaly detection methods to these datasets to identify trends. Our initial results show that the algorithms exhibit the expected performance trends with respect to the data set properties. This shows that our data set construction methodology is producing sets where the properties are meaningful. Further the results indicate that the isolation forest algorithm is particular effective and robust, though this is an early result that requires further investigation.

We are currently expanding on this initial study in two ways. First, we are rebuilding the benchmark data sets to incorporate several minor improvements including a baseline setting in which no manipulation of the dimensions is performed. Second we are including more anomaly detection algorithms in the empirical evaluation.

Research in ensemble methods for anomaly detection

Ensemble methods have been extremely successful in supervised learning and clustering. It is important to distinguish between homogeneous ensembles and heterogeneous ensembles. In a homogeneous ensemble, a single algorithm is executed multiple times (with injected randomness) to produce an ensemble. The EGMM and IFOR algorithms are good examples of this, and ensemble methods are very important to their good performance. A heterogeneous ensemble combines a diverse set of algorithms in an attempt to achieve robust performance across a wide range of problems. In supervised learning, heterogeneous ensembles typically involve training a "stacked" classifier that computes a weighted combination of the members of the ensemble. The training signal for this stacked classifier is provided by various hold-out techniques.

The leading heterogeneous ensemble method for anomaly detection was developed by Schubert et al. [Schubert 2012]. It forms a consensus labeling of the data points (as anomalous or normal) using a simple consensus method in which each of the anomaly detectors in the ensemble is allowed to vote for K data points. Any point that receives a vote is labeled as anomalous. These consensus labels are then applied to learn weights for a stacked anomaly detector.

Oregon State undertook extensive algorithm development and experimentation on methods for creating heterogeneous ensemble for anomaly detection with the goal of improving upon the Schubert method. Here is a summary of our findings:

1. Most anomaly detectors agree on the anomalies; therefore, simple aggregation of ranks or scores (min, max, avg., median, geometric mean) strengthens signals from true anomalies. Geometric mean appears to be the most reliable and has some theoretical justification.

2. Sophisticated methods that model the empirical rank/score distributions more accurately and preserve most anomaly signals (e.g., mixture of Gaussians, mixture of Gammas, Bayesian aggregation of Rank Data (BARD; [Deng 2014]), and Plackett-Luce (PL; [Plackett 1975])). The mixture approaches did not show any AUC improvement even in cases where the score distributions were satisfactorily inferred. BARD was able to infer detector quality quite well, but its results are almost at the median of all detectors. PL, which uses straightforward rank aggregation, performed much worse than most base algorithms.

3. Multiple types of anomalous patterns exist which leads to broad disagreement across specialized detectors. Simple aggregation of ranks or scores suppresses some of these patterns. A mixture model (such as rank-based topic models or mixtures of PL distributions) might instead untangle those patterns and report anomalies from each. It is however difficult to check if each mixture component corresponds to a particular pattern in real-world high dimensional data because the number of anomalies is typically small. In our experiments, evaluation did not show any improvement in AUC over the top-performing base algorithms.

4. One possible approach to learning ensemble weights would be to request and integrate feedback from the analyst. We developed two methods for employing expert feedback, one for the BARD algorithm and one based on learning weights using the Passive Aggressive (PA; [Crammer 2006]) algorithm. Feedback with BARD increased the variance in results and showed no overall improvement. The PA algorithm shows some promise. Two fundamental problems which need to be solved are (a) class imbalance which leads to almost one-sided feedback, and (b) inferring a decision boundary so that uncertainty sampling (shown to be very effective in active learning) might be used as a query strategy. This is where we are focusing our current research. **Figure 4** and **Figure 4** are two results that use random projection vectors from the LODA algorithm [Pevný 2016] as the ensemble components. The horizontal axis reports the number of cases for which the analyst has provided feedback. The vertical axis is the AUC. The dashed line is the AUC of LODA without ensemble reweighting, and the red curve reflects reweighted via the PA algorithm. In both of these benchmark problems, reweighting is able to achieve a small but significant increase in AUC.

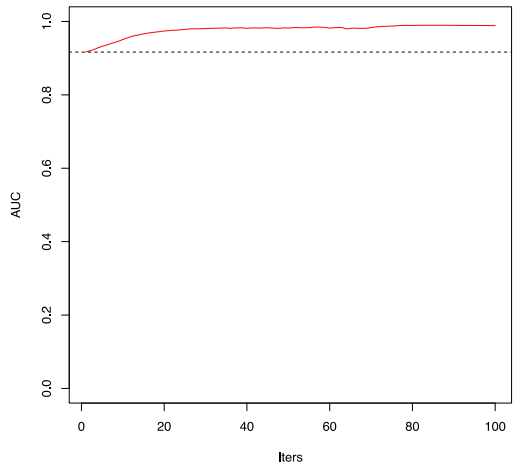


Figure 6. *Forest Cover*

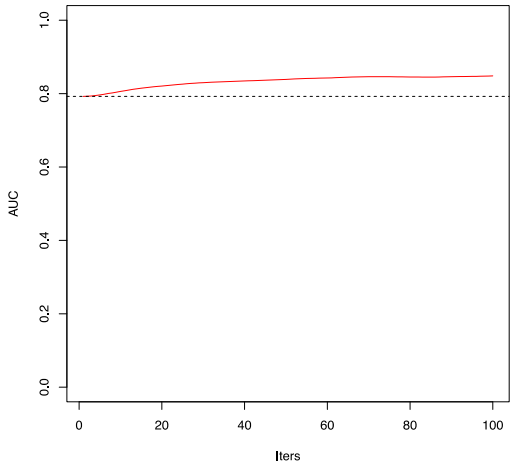


Figure 5. *Particle*

1.3.2. Research in anomaly explanations

Oregon State conducted extensive research on methods for explaining anomalies to analysts. We studied a particular form of explanation which we call a *Sequential Feature Explanation* (SFE), which consists of an ordering over the features describing the anomalous case. The anomaly detection system explains why it has marked a case as anomalous by presenting the features in order until the analyst has enough information to make a decision about whether to open an investigation of the case. We imagine a user interface in which the analyst is shown the value of the first feature for the case (in the context of the value of the first feature for other cases). The analyst may be able to immediately see why the point was marked as anomalous. If the analyst is still not convinced, then the combination of the first and second features is presented (in context). If this is still not sufficient to allow the analyst to make a decision, the first three features are shown. Over time, we expect the analyst to become more and more certain that the point is (or is not) an anomaly. We summarize this as an Analyst Certainty Curve (see **Figure 7**). The horizontal axis records the features as they are incrementally revealed to the analyst. The vertical axis is the probability that the analyst assigns to the point being "normal".

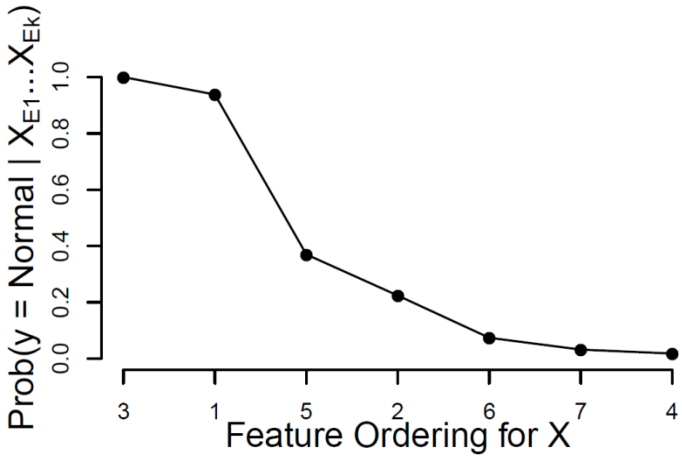


Figure 7. *Analyst Certainty Curve*

The goal of our explanation algorithms is to minimize the number of features that must be revealed to the analyst. We refer to this as the Minimum Feature Prefix (MFP).

We developed four algorithms for constructing sequential feature explanations and compared them to randomly-ordered features and to an oracle that orders the features optimally. On benchmark problems, we create a simulated analyst by applying supervised learning to train classifiers to predict the probability that the data point is "normal". We set a probability threshold (e.g., 0.3) below which the explanation process is terminated.

Experiments showed that the best algorithm for SFEs is the Sequential Marginal algorithm, which is a form of greedy forward selection. In this work, we employed the EGMM anomaly detector, so we assume that it has computed a density estimate $f(x)$ over the data space. The Sequential Marginal algorithm computes the feature j_1 that minimizes the marginal density $f(x_{j_1})$ and makes feature j_1 the first feature in the sequential feature explanation. It then finds the feature $j_2$that minimizes the pairwise marginal $f(x_{j_1}, x_{j_2})$, and makes j_2 the second feature in the explanation. This greedy selection process continues until all features have been placed into the SFE.

Figure shows the performance of Sequential Marginal (SeqMarg) compared to Random and to an Oracle (OptOracle). We observe that the Random method requires many more features than the Sequential Marginal method. In some cases, the Sequential Marginal method comes close to matching the Oracle, but in other cases, it requires twice as many examples.

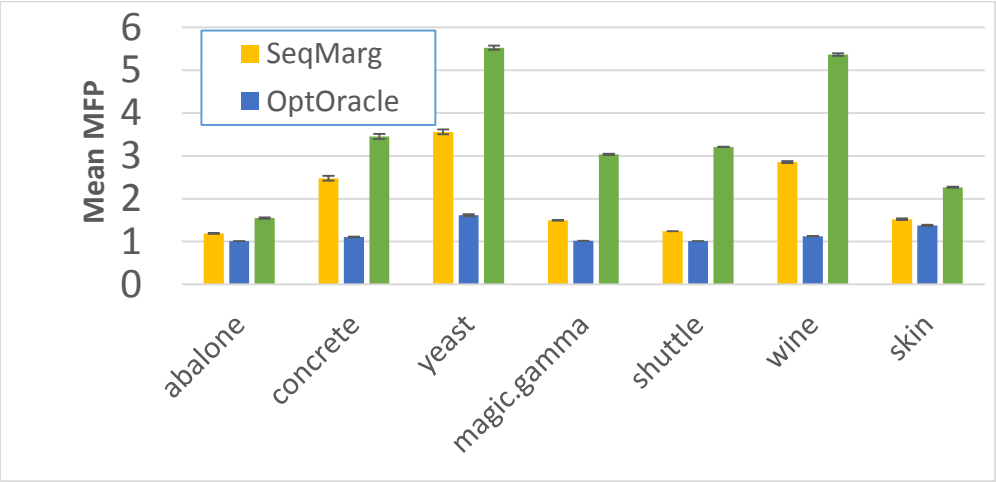


Figure 18. *Performance of Sequential Marginal (SeqMarg) compared to Random and to an Oracle (OptOracle).*

We studied additional algorithms that optimize the MFP globally instead of using a greedy algorithm. However, these algorithms did not provide a noticeable improvement over Sequential Marginal despite their much greater computational cost. This work was presented at the KDD 2015 ODDx3 Workshop [Siddiqui 2015].

1.4.University of Massachusetts Algorithm Research Activities

Research at the Knowledge Discovery Laboratory at the University of Massachusetts Amherst focused on developing several novel algorithms. However, this work also produced evidence for several general conclusions about the task of anomaly detection for insider threat analysis as well as core concepts commonly used for anomaly detection. These conclusions included the following:

Anomaly detection as joint probability estimation — Anomaly detection for insider threat analysis, as well as many other anomaly detection tasks, can be accurately defined as identifying data instances with low joint probability and can be solved by applying high-accuracy joint probability estimation. This conclusion appears obvious to some investigators, but is far from universal in the existing literature on anomaly detection.

Classifier-adjusted density estimation (CADE) — Remarkably efficient and effective anomaly detection can be performed using combinations of very simple joint probability estimators and a single conditional probability estimator. We implemented and evaluated a variety of such estimators, but the basic concept is quite simple: A very simple generative model is defined that approximates the joint probability distribution, and then a conditional probability estimator is used to "correct" the probability estimates of the simple joint estimator. This approach has been previously proposed, but had not been widely used or empirically evaluated.

Normalization — Normalizing features generally improves anomaly detection for insider threat analysis, although no single normalization approach was uniformly superior. We evaluated a large number of normalization approaches, and we defined a general framework for describing such approaches.

Mixtures of dependency networks — Significant improvements in joint probability estimation can be obtained by learning latent groups of data instances. Our approach to learning this latent structure (mixtures of dependency networks) obtained significant improvements over single dependency networks or a range of other joint probability estimators.

Feature importance — Some information useful for explaining anomalies can be provided by joint probability estimators such as dependency networks. A robust and relatively simple approach to explaining anomalies is to identify the features that, if they had an alternative value, would have greatly increased the joint probability of a given data instance (e.g., a user-day). Such features are those that have the greatest individual impact on whether a data instance is deemed an anomaly.

1.4.1. Research in joint probability and causal reasoning models (UMass SOW 1.1.4)

UMass-1: Classifier-Adjusted Density Estimation (CADE)

We developed a class of estimators for joint probability distributions, often referred to as *density estimators*, that are both highly accurate and computationally efficient to learn and apply. This class of estimators, which we refer to as *classifier-adjusted density estimation* (CADE), overcomes key disadvantages of existing methods for joint probability estimation. Many of the current joint probability estimators that are in wide are either: (1) based on unrealistic assumptions about the domain that impair their accuracy (e.g., feature independence); or (2) computationally inefficient to learn and apply. In contrast, CADE-based estimators make few assumptions and are efficient to learn and apply.

As we note above, joint probability estimators are directly useful for anomaly detection. If the joint probability of a set of features can be accurately estimated, then data instances (e.g., user-days) whose features have low joint probability are by definition anomalous. Such instances don't necessarily represent insider threats (or any other specific underlying cause of anomalousness), but they are excellent candidates for further investigation.

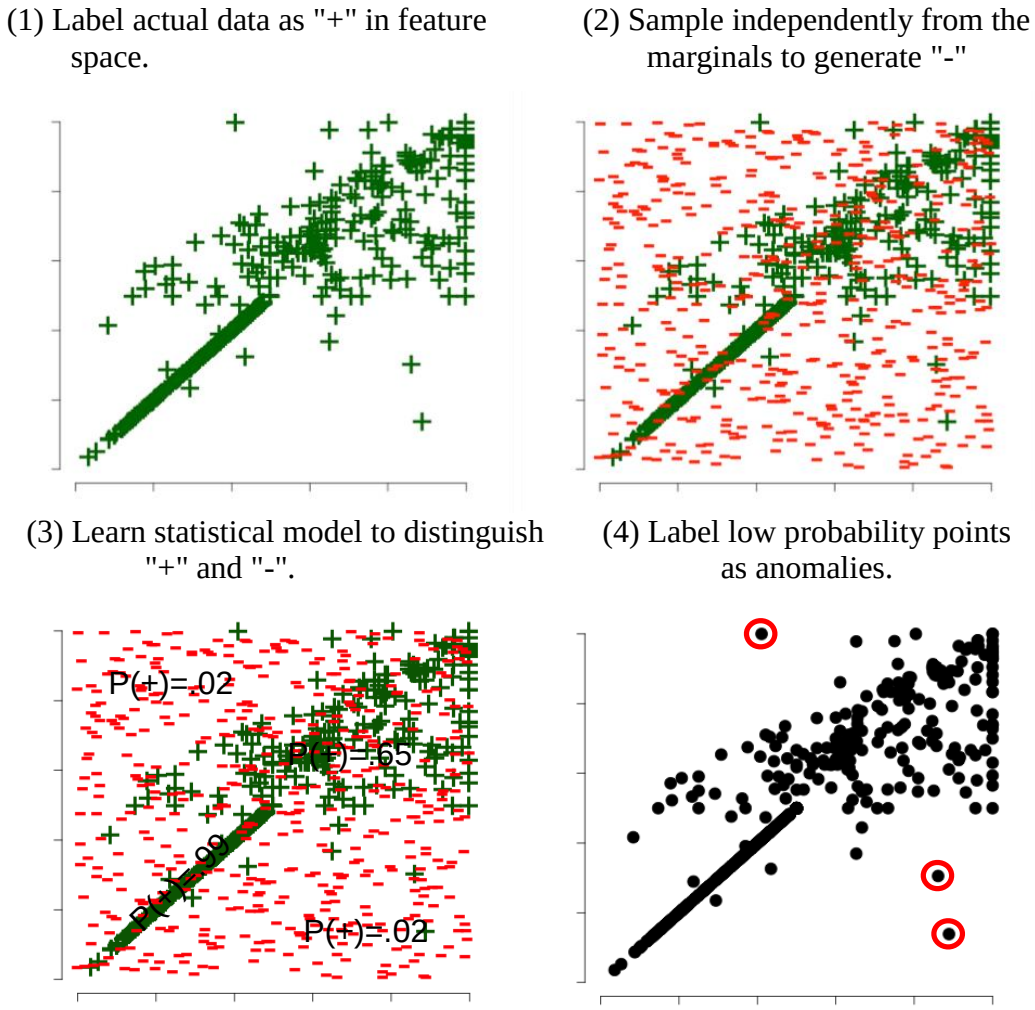


Figure 8. *Detecting anomalies using CADE*

CADE constructs a model of normal behavior by first taking a sample of the observed data instances, treating them as non-anomalous, and constructing an equal number of pseudo-anomalies (**Figure 8**). Some initial joint probability estimator is constructed from training data and then used to generate synthetic data points that we call "pseudo-anomalies." This estimator often makes unrealistic or unlikely assumptions about the nature of the joint distribution (e.g., feature independence) in the interests of efficient learning and inference. CADE then constructs a conditional probability estimator (a "classifier") to distinguish the observed data instances from the pseudo-anomalies.

When given a new instance, CADE combines the classifier's prediction with the initial joint estimator's distribution to determine whether the instance is anomalous. This approach produces a representation of the joint distribution that is sufficient for anomaly detection and that is highly efficient to learn and apply. For example, when using unoptimized code on commodity hardware, CADE learns the classifier and assesses all entities in less than 10 minutes on a data set of 131,729 entities with 83 features, and it outputs a score corresponding to the degree of anomalousness of an entity extent.

In the course of our work and refinement of the ideas behind CADE-based models, we discovered that the ideas behind CADE have been re-discovered several times, but the properties and performance of CADE-based models have not been widely understood. Hastie et al. [Hastie 2001] presented this idea more than a decade ago with the still-appropriate remark that although it "seems to have been part of the statistics folklore for some time, it does not appear to have had much impact despite its potential to bring well-developed supervised learning method- ology to bear on unsupervised learning problems." Our research under ADAMS was the first to compare a wide range of alternative specific technologies for the two components of CADE (a simple joint estimator and a conditional estimator), and our work was the first to compare its performance to several other candidates.

We showed that several combinations of existing density estimators and classifiers can produce high-quality joint probability estimates, although some combinations perform quite poorly and should be avoided. Our research also demonstrated that these density estimation methods scale well to data with high dimensionality and that they are robust to the problem of irrelevant attributes that plagues methods based on local estimates. In addition, we showed that CADE is surprisingly robust to the addition of random "noise" attributes, particularly in comparison to another widely used method for anomaly detection ("local outlier factor"). Full details are given in a conference paper describing and evaluating CADE [Friedland 2014].

UMass-2: Relational Density Estimation (RDE)

In addition to CADE, and largely for experimental comparison to CADE, we examined, implemented, and evaluated additional classes of density estimators. Our goal was to support greater analysis and explanation of why particular instances were identified as anomalous, as well as learning and reasoning with the relational structure of the Vegas data.

For example, one of these models assumed feature independence but focused on high-accuracy modeling of the marginal distributions. Each marginal distribution is modeled using a kernel density estimator, and the joint probability is assumed to be a simple product of these marginal distributions. The probability estimates themselves are biased when the independence assumptions are violated, although rankings can still be accurate despite these biases. While these methods were efficient to construct and apply, their performance was generally below that of CADE.

Propositionalization and Normalization techniques

As part of our research and data analysis, we realized that substantial improvements in anomaly detection could be realized by transforming data in two ways. First, we applied relatively standard methods to convert between the multiple data tables available from Vegas (including inter-related users, machines, websites, email messages, etc.) and the single table of data necessary as input to most classifiers and density estimators. This process, often called "propositionalization" converts multiple tables to a single table.

Second, we "normalized" data in several ways that minimize irrelevant variability in data values and more clearly highlighted individuals and days that vary with respect to a group of other individuals or days. Specifically, we explored a range of feature normalizations: median-difference and percentile of all users on all days in time period; median-difference and percentile of all the user's days in time period; median-difference and percentile of all users on same day;

median-difference and percentile of all users in group i on the same day; and median-difference and percentile of all users in group i's days in a given time period.

Intuitively, for a given feature: normalizing each day's value to all the user's days should highlight unusual days; normalizing each day's value to all users on that same day should dampen day-to-day, across-the-board variations; normalizing each day's value to a user's group should heighten variation from peers. Unlike median-difference normalization, percentile normalization is insensitive to distant outliers.

We found that percentile normalization by each user's days gave the best results on the Vegas data. This was, at some level, surprising. Prior to obtaining these results, we had assumed that normalizing by workgroups or by all users in a time period would have been the most successful. The former case could have removed variability with respect to the behavior of peers, and the latter case could have removed variability associated with weekly, monthly, and yearly events. However, per-person variability appears to be the most important effect to represent and account for in normalization.

1.4.2. Research in causal dependence in complex domains (UMass SOW 2.1.4)

Dependency Networks and Mixtures of Dependency Networks

Dependency networks (DNs) are a class of highly accurate joint probability estimators that are very efficient to learn [Heckerman 2001, Neville 2007]. Each DN consists of *k* conditional probability estimators (one per variable) that are learned independently. The structure of a DN indicates marginal dependence among variables, and can be quite sparse, particularly when selective conditional estimators are used for learning (**Figure 20**).

A single dependency network offers an efficient and flexible method for modeling a joint distribution. However, for complex distributions with large numbers of variables, single dependency networks may fail to provide the necessary representational power. Specifically, using a single dependency network model to represent a complex distribution can lead to problems in inference efficiency and interpretability.

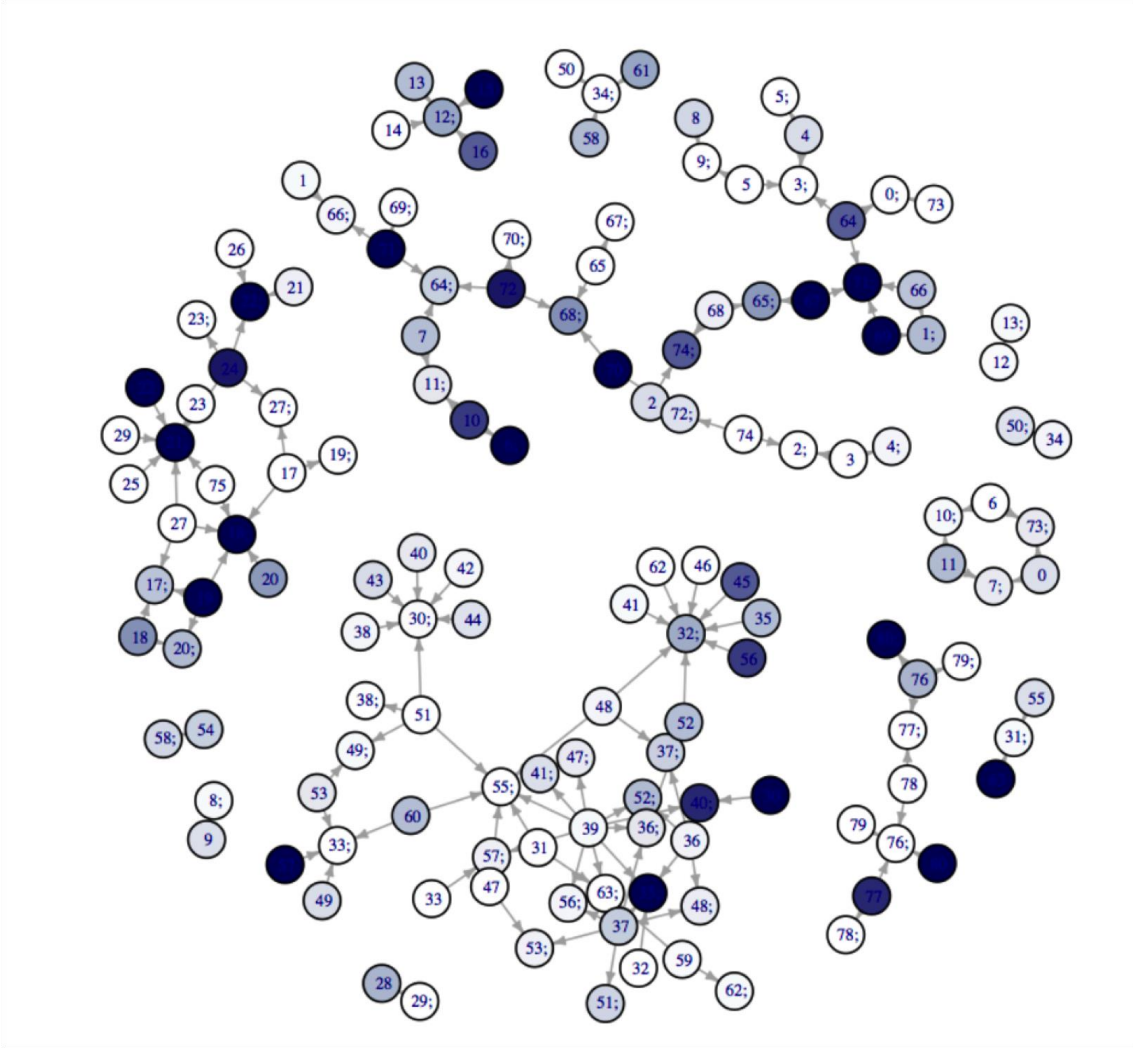


***Figure 20**. A dependency network (DN) for the September 2012 Vegas data learned using classification trees as the conditional estimator (alpha = 0.001). DNs can be used to estimate per-instance feature significance h by estimating the conditional log-likelihood of individual feature values (most anomalous insert, darker = higher).*

Inference efficiency can be impaired because the quality of the approximation to the likelihood function is known to depend on sparsity of the underlying model for convergence rates. As the number of variables and the complexity of the underlying joint distribution grow, the sparsity of the dependency network representing it decreases. This is especially the case for distributions that contain instances of context-specific independence (CSI). CSI describes a case in which two variables are dependent in some contexts, but independent in other contexts, where "context" usually refers to the value of some set of additional variables.

Interpretability can be impaired because a single dense dependency network can be very difficult to interpret graphically. This is especially true in the case of CSI. First, the existence of an edge between two variables implies that some dependence between the variables holds in all cases, even if the details of the conditional probability distributions imply that such dependence holds only for specific values of other variables. Second, the dependency network will also imply that

the context-setting variables have a dependence with other variables, even though their role is merely to determine the context of other dependencies.

To overcome these limitations, we developed the *mixtures of dependency networks* (MDN) framework shown below in Figure 21. MDNs follow the intuition of earlier work in mixtures of graphical models that, rather than attempting to model a single, dense network, induce sparsity by learning a mixture where each component is represented by a graphical model. MDNs distinguish themselves from other forms of graphical models by not relying on strong constraints about the dependence structure (as is the case in the mixture of trees) or expensive learning procedures for each component (as is the case in learning mixtures of Bayesian networks).

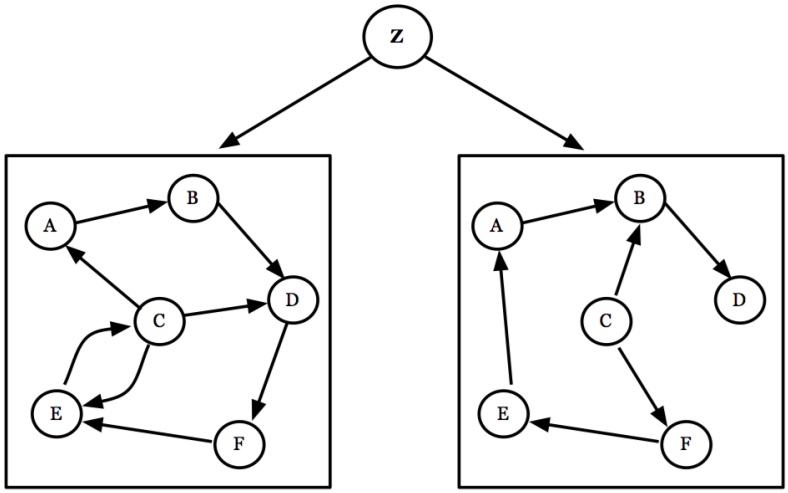


Figure 21. *An example of a mixture of dependency networks. The hidden variable z denotes membership to each component. Each component is a fully formed dependency network.*

MDNs are particularly appropriate for the data sets considered in ADAMS, where joint distributions are unlikely to be drawn from a single generating function. Instead, data sets are likely to be mixtures of many different generating functions. To better model this, we used MDNs to learn sets of dependency networks that model such heterogeneity automatically. We conducted experiments on a wide range of data sets that showed that MDNs systematically outperform single dependency networks and the three other state-of-the-art density estimators evaluated: mixture of trees (MOTs), sum-product networks (SPNs), and arithmetic-circuits Markov networks (ACMNs). These results are presented in detail in a paper under submission [Arbour 2016].

Feature sensitivity analysis

Providing analysts with information about what caused an individual instance to be scored as anomalous ("explanation") was an important objective of Phase 2. Our early, Phase 1, ADAMS models performed well, but they produced opaque joint probabilities that were used to score individual instances (e.g., user-day instances scored by CADE). A challenge in Phase 2 was to continue to use these effective models, but to develop techniques that could determine which features contributed most to an instance's score in order to aid explanation and investigation.

Our approach to this explanation challenge was to measure how much the instance's anomaly score would drop as feature values were adjusted slightly in the direction of "normal." Specifically:

- For each feature, create a variant instance where the original feature value is changed towards "normal" (e.g., highest conditional probability)
- Use the joint probability model to score each of the instance's feature variants and record the change from the original instance's score
- Normalize the sum of score changes to 1.0, rank the features by descending change, and present the top-ranked estimates of feature importance

These feature-importance estimates could be computed on demand for individual instances or in batch (e.g., on the 200 most anomalous instances) in response to analyst needs.

Insider Threat Detection Research & Results

1.5.Counter-Intelligence Domain Analysis (Leidos SOW 1.2.1)

In Phase 1 we applied domain knowledge to the analysis of cyber observables to form patterns of activity that indicate known or suspected insider threats is central to our research. Our hypothesis was that these patterns would serve as starting points for making inferences about user behaviors based on detected anomalies using the semantics (i.e., the attributes) and the structures (i.e., the graphs) in the data. We categorized these patterns in three ways – indicators, anomalies, and scenarios, and each entails aspects of domain knowledge. Indicators are patterns of user behavior found in cyber observables that correlate with malicious insider actions as derived from case studies (e.g., abnormal removable media activity indicative of IP theft). Anomalies focus on unusual patterns in the data (e.g., logins after hours, large numbers of file events on network drives). The domain knowledge applied in this type of experiment draws on an understanding of the data as it relates to human-computer interactions and how abnormal patterns from a malicious insider might occur in cyber observables within single or multiple feature sets over different time ranges. Scenarios describe complex patterns of malicious insider actions that span data types (e.g., email, file access, login, and URL) and entity, population, and temporal extents and baselines. Examples of scenarios include Saboteur and IP Thief. We believe that these three starting points would enable top-down (i.e., domain knowledge-driven) and bottom-up (i.e., data-driven) experimentation.

Our first step in the application of domain knowledge as part of ADAMS research entailed the identification of personality traits and workplace behaviors frequently associated with malicious insiders based on inputs from a subject matter expert and historical case studies. Our subject matter expert, a retired intelligence operations officer, developed a list of 182 demographic descriptors (e.g., age ranges; education level; passed over for promotion or marginalized), personality traits (e.g., a sense of superiority in relation to peers; disassociation with the organization); and workplace behaviors (e.g., a disregard for security policies; unexplained absences). **Table 1** provides examples of these traits.

Table 1. *Examples of traits of personality traits from domain knowledge for ADAMS.*

Shares private accounts and passwords with others or requests that others share private account information with them	Request and/or obtains exceptional local or network privileges that are beyond assigned job duties (e.g., an intelligence analyst who asks for system administrator privileges)	Use “backdoor” accounts to access computers or networks without permission	Anti-virus software is perpetually out-of-date; manually disables automatic updates	Disables event logging, modifies firewall settings, removes anti-virus software without permission
Frequently connects personal devices to their work PC and transfers files	Participates in discussion forums, chats, and newsgroups using their company e-mail address	Manipulate system logs on workplace computers (e.g., delete personal history log of websites visited)	Access machines on other networks (e.g., a home network) that circumvent company security measures to do	Use an unauthorized device (PDAs, flash drives) to gain access to a system or network

			personal tasks	
Frequently uses work computer/network for social media activity	Frequently accesses unsecured wireless networks	Gains unauthorized access to an internal or external system by bypassing established protocols	Uses corporate computer, network, or other resources (e.g., email account) to post inflammatory or derogatory material about an individual or group	Conveys information about personal distress (health, family, finance) from a work computer, network, or other resource (e.g., email address)
Plays unauthorized/not approved web-based games using corporate systems during core business hours	Conducts vulnerability research about specific computer systems or networks that is not part of assigned job duties	The employee expresses sentiment that he/she is under used/under employed	The employee is repeatedly passed over for promotion, or career progression lags behind peers (experience, skill level, demographic)\	Uses computer resources (to include internet connection) to conduct unauthorized or unapproved secondary employment/work activities

We also classified malicious insider behavior into five stages of activity that addressed three broad categories of motivators, or goals. We defined these as: Exploration; Experimentation; Exploitation; Exfiltration; and Escape and Evasion. The goals we identified were: theft or misuse of data or systems; corruption of data; and destruction of data or systems. **Figure 9** relates the stages of malicious insider activity to a motivation or goal.

Motivation / Stage	Theft of data or misuse of systems	Corruption of data	Destruction of data or systems
Exploration	*Locate data*	*Locate access points*	*Locate weak points*
Experimentation	*Trial access*	*Trial modifications*	*Trial insertions*
Exploitation	*Staging data*		
Execution	*Exfiltrate data*	*Modify data*	*Plant malware*
Escape & Evasion	*Layer movement*	*Cook books*	*Shift blame*

Figure 9. *Insider Scenario – Motivations and Stages*

Once we had defined the stages of behavior and motivations for each stage, we listed specific, cyber-related actions that occur at each stage. We called these cyber-related actions indicators. Examples of indicators related the Exploration stage of malicious insider behavior include: mapping shared drives; listing network administrators; pinging protected servers; probing

protected files; and a user altering his/her accesses. **Figure 10** lists the full list of indicators we developed. Note that we developed our set of indicators to be general to malicious insider activity outside the Vegas dataset; therefore, we did not anticipate that all indicators would apply to the full range of experiments performed in Vegas.

1. Exploration
1.1. Mapping Shared Drives
1.2. Listing Network Administrators
1.3. Pinging
1.4. Probing protected files
1.5. Access alterations
2. Experimentation
2.1. Attempted logins from a variety of machines and locations
2.2. Attempted logins with improper user ids or passwords
2.3. Frequent remote logins from disparate locations
2.4. Access (or attempted access) which is inconsistent with the user's data constellation (think Multiple User Accounts)
2.5. Manipulation of file systems
2.6. Unauthorized communications protocols (proxy servers, firewall subversion, etc.)
2.7. Data transport alterations (internet versus intranet)
2.8. Creation of virtual drives (encrypted, hidden)
2.9. Disable anti-spyware software on your own computer
3. Exploitation
3.1. Frequent and large downloads (grab and go)
3.2. Unauthorized Data encryption
3.3. Frequent external emails
3.4. Unusually large storage devices
3.5. Unauthorized software installations (screen capture software, Google Desktop, Warez phenomena, etc)
3.6. Installed automated processes
3.7. Hard Drive Imaging
3.8. Hidden Virtual Disk Images
3.9. Accept invalid SSL certificates
4. Exfiltration
4.1. Use of personal storage devices
4.2. Downloading digital media to disks or storage devices "to take home"
4.3. Unexpected "Working from home" (remote connections into the work site)
4.4. Encrypted data transmissions outside the authorized domain
4.5. Unusually timed remote system access
4.6. Access unsecured or non-encrypted wireless networks
4.7. Keep unused ports open and accessible to the web
5. Escape & Evasion
5.1. Deleting Log Files
5.2. Unauthorized Database Encryption
5.3. Reformatting drives

Figure 10. *Indicators*

After we had developed the list of indicators, we inserted them in patterns of activity consistent with each PRODIGAL scenario (i.e., Saboteur; Intellectual Property Thief – Ambitious Leader; Intellectual Property Thief – Entitled Individual; Fraudster; Careless User; and Rager). **Figure 24** summarizes the PRODIGAL scenarios. Note that the gray font used for the Careless User and Rager scenarios indicates that we did not address fully these scenarios in Phase 1 as they extensively used unstructured text data and semantic and content analysis approaches, which we did not emphasize as part of our Phase 1 research thrusts.

Title	Description
Saboteur	CERT*: An insider's use of information technology (IT) to direct specific harm at an organization or an individual. Saboteurs are technical, such as a system administrator, have privileged access to systems, revenge is generally their motivation, they set up their attack before termination, and they execute the attack after leaving.
Intellectual Property (IP) Thief – Entitled Individual	CERT: An insider's use of IT to steal IP from the organization. This category includes industrial espionage involving insiders. IP thieves are generally scientists, engineers or salespeople. They generally steal what they consider to be their own work from their former or current project and use it to start a new company or give it to a new company or foreign organization.
IP Thief – Ambitious Leader	CERT: An IP thief that steals information like the EI, but is motivated by ambition to steal as much as possible before leaving the organization, rather than dissatisfaction or a dispute. To do so, he recruits other insiders to get access to all parts of the IP being stolen.
Fraud	CERT: An insider's use of IT for the unauthorized modification, addition, or deletion of an organization's data (not programs or systems) for personal gain, or theft of information that leads to an identity crime (e.g., identity theft, credit card fraud). Fraudsters are lower-level employees, are often motivated by financial need - hardship, greed, etc., they sometimes are recruited by outsiders in collusion with other insiders.
Careless User	PRODIGAL: The insider is not intentionally malicious but, through blatant disregard of corporate policies concerning IT systems, exposes the group to a comparable level of risk (i.e., compromising systems and data) similar to the Saboteur scenario.
Rager	PRODIGAL: The insider has outbursts of strong, vociferous, abusive, and threatening language in Emai/Webmail/IM repeatedly toward other insiders or against the organization in general . These outbursts coincide with anomalies in other data types, e.g., Logons, URL, indicating a potential fundamental change in behavior.

*** Description adapted from: Capellli, Moore, and Trzeciak. The CERT Guide to Insider Threats: How to Prevent, Detect, and Respond to**

Figure 11. *Insider Threat Scenarios*

We then aligned indicators with each scenario and analyzed the potential data types with each indicator. See **Figure 25.**

Scenarios / Indicators	1: Planning Attack	2: Escalating sleeper	3: Gradually goes Bad	4: Unintentional causes	5: Manipulated Individual	6: Manipulated Group	7: Hidden Group
1.1: Maps drives	X	X	X				X
1.2: List administrators							
1.3 Pings	X	X	X		X		X
1.4: Probes files	X	X	X	X	X	X	X
1.5: Alters accesses	X	X	X	X	X	X	X
2.1:Roving logins	X	X	X	X	X	X	X
2.2: :Failed logins	X	X	X	X	X	X	X
2.3: Remote logins	X	X	X	X	X	X	X
2.4:Role-privilages	X	X	X	X	X	X	X
2.5: Files manipulated	X	X	X	X	X	X	X
2.6: Improper comms	X	X	X			X	X
2.7: internet vs intranet	X			X			
2.8: Modifies drives	X	X	X	X	X	X	X
2.9: Disables security		X	X		X	X	X
3.1: Download activity	X	X			X		X
3.2: Modes encrypted files		X			X	X	X
3.3:Emails externally		X			X	X	X
3.4: Uses external storage						X	
3.5:Installs software	X	X	X	X	X	X	X
3.6: Installs processes	X	X	X	X	X	X	X
3.7: Images hard drive	X	X	X	X	X	X	
3.8: Hides virtual disks	X	X	X	X	X	X	
3.9: Accepts certificates	X	X					
4.1: Use personal devices							
4.2: Downloads remotely							
4.3:Remote connections	x	X	X	X	X	X	X
4.4: Encrypts outside SLAN	X	X	X	X	X	X	X
4.5: Remote system access	x	X	X		X	X	X
4.6: Access wireless LAN	X	X	X	X	X	X	X
4.7: Opens unused ports		X	X		X	X	
5.1: Deletes log files	X	X	X	X	X	X	X
5.2: Encrypts		X	X	X	X		

***Figure 12**. Aligned indicators with each scenario*

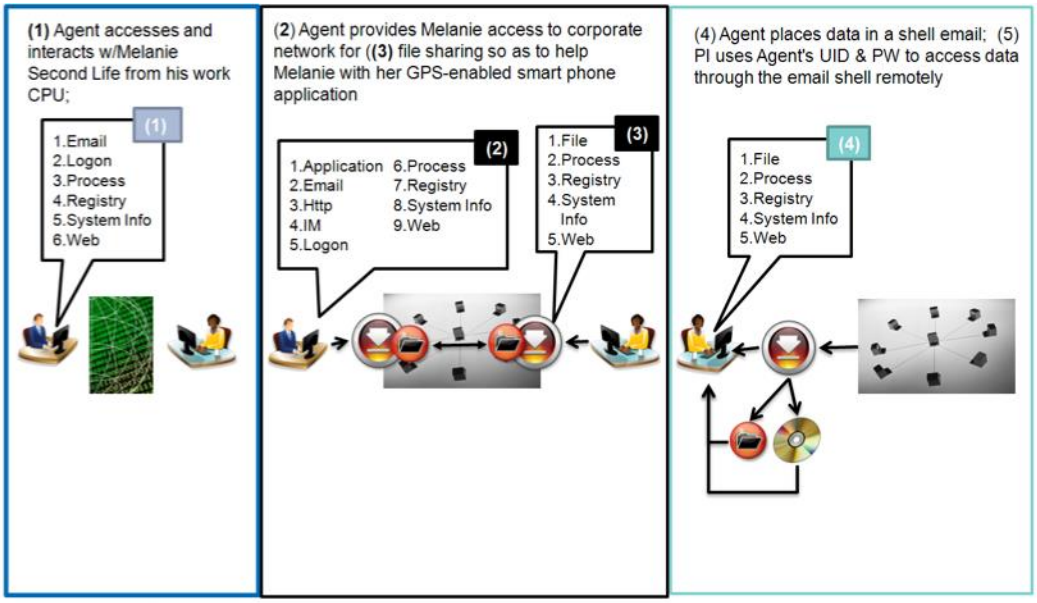


***Figure 13.** Insider threat actions and potentially associated data types.*

When the Vegas data became available in Year 2 of the program, we performed exploratory analysis of the data types available for ADAMS (e.g., File, URL, Logon, Process, IM, Email, and Registry). We also assessed the suitability of the data for PRODIGAL indicators and scenarios. We projected that the Vegas data could support experiments featuring twelve of the 33 indicators and all six scenarios. For details on PRODIGAL experimentation, see section 1.8.

Phase 2 insider threat domain analysis research concerned the analysis of PRODIGAL's performance in the context of the Red Team scenarios, the presentation of PRODIGAL results to the analysts in domain terms, and preliminary research into the feature enrichment using attributes from Vegas data.

For each dataset – over 90 in total – we decomposed the Red Team user behaviors in the context of the plain-language descriptions of the scenario and the inserted observations (for each Red Team user in the scenario) that represented the insider threat actions simulated in the data. We used this scenario decomposition to support post-mortem analysis of PRODIGAL results by dataset and scenario (as there were multiple instances and variants of scenarios over multiple months' of data). Domain analysis characterized the types of malicious insider behaviors encompassed by the various scenarios, informing the researchers as to the relative difficulty of detecting insider threats, given the presence of specific combinations of observables.

Figure 27 is an example of the domain analysis performed, relating the specific malicious insider action from a scenario (exfiltrating sensitive information using removable media) to an observable-based workflow. Scenario decomposition at the user activity-level identified crucial observable types from the domain perspective.

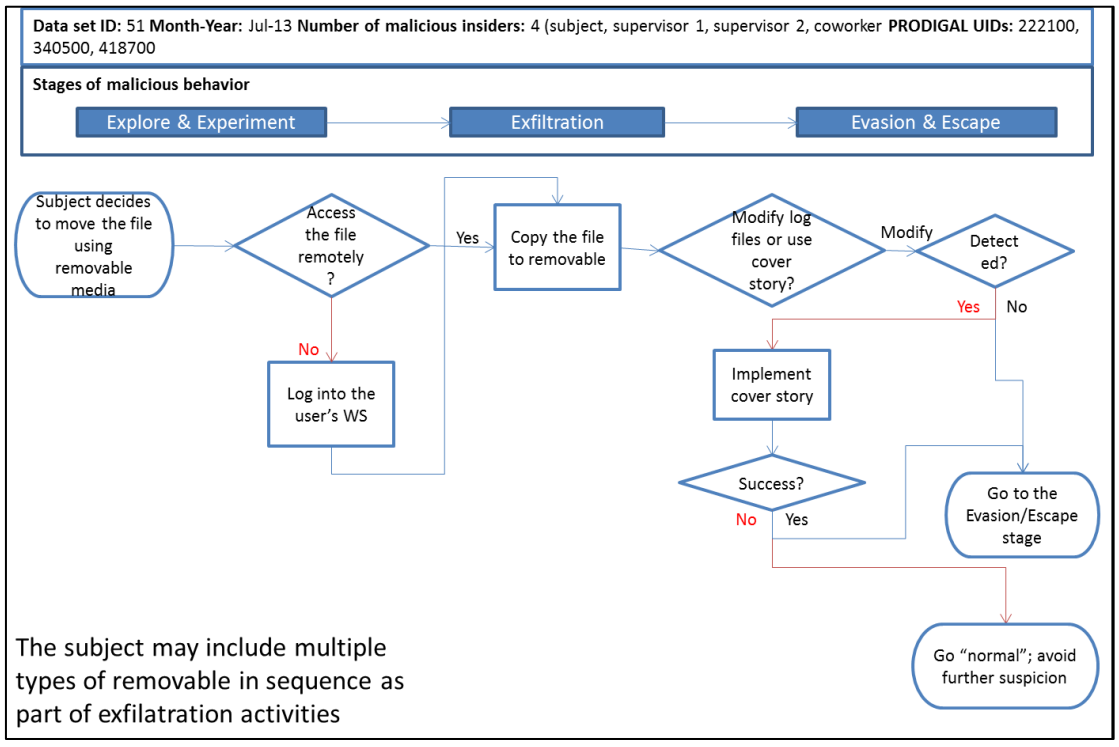


Figure 14. *Example of exfiltration behavior: copying to removable media.*

- Subject has continued with research concerning Bradley Manning, Bradley Manning connected to WikiLeaks, and Bradley Manning's treatment by the U.S. Government.
- Subject has engaged a co-worker concerning Bradley Manning. Subject has been researching whistleblower laws, sites that accept and post whistleblower material, and confidential reporting mechanisms.
- Subject has created text files and batch files that export segments of the text files through specially crafted DNS queries.

Figure 15. *Example of the post-mortem scenario analysis. Manning up redux – answer key 1 of 2 March 2013.*

The following diagrams are examples of the post-mortem scenario analysis we performed to support the analysis of PRODIGAL results. We began with plain-language description of the scenario (as summarized from the Red Team's narrative).

We then re-constructed the user's malicious activities in the context of the scenario description, by data type and day, for each data set. **Figure 29** summarizes the red team user activity for a "Manning Up Redux" scenario (dataset 36) in which a malicious insider who is a system administrator uses removable media to steal information.

- Number of RT users and the total number of inserted observations: 2 users, 1088 inserts
- Total number of users and ensemble scores for the month: 5721, 177291
- Data types of inserted observations: URL, Process, File, Keyboard
- Date range: 17 user/30 calendar days from 3/1/13 to 3/29/13
- Total number of observations (and inserts) by data type for the RT user

User ID	total	Logon	File	Process	URL	IM	Email	Keyboard	Printer
462300	1028	249 (0)	606 (12)	1679 (461)	804 (530)	869 (0)	3312(15)	368 (10)	0 (0)
547100	60	179 (0)	2251 (0)	493 (45)	45 (0)	1699 (0)	7601(15)	1254 (0)	0 (0)

- Inserts by user, calendar date

User ID	1-Mar	4-Mar	6-Mar	7-Mar	8-Mar	11-Mar	14-Mar	15-Mar	18-Mar	19-Mar	20-Mar	21-Mar	26-Mar	27-Mar	28-Mar	29-Mar
462300	51	74	18	32	263	48	102	24	8	0	138	9	118	84	**59**	0
547100	0	0	0	0	0	34	0	0	0	13	0	0	0	0	0	***13***

- (*Note: Bold, italicized font indicates the insert count on the highest-ranked user day for a given RT user; bold underlined font indicate the insert count on the highest-ranked RT user day for the month)*
- ***Highest rank for a RT user in March: 76 out of 177291 (user 462300 on 3/28/13)***

Figure 16. *Example of the post-mortem scenario analysis. Manning up redux – answer key 2 of 2 March 2013.*

In Phase 2, to enhance the effectiveness of PRODIGAL results in the context of analytic use cases, we also performed domain analysis to inform our explanation research. The objective of our explanation research was to the ensemble user-day scores, the primary output of the system, to the analyst in the context of domain terms. Intuitive explanations support an analyst's

decision-making processes when interpreting a user's behaviors in terms of specific, discrete activities on their computers. When analyzing user anomaly scores, explanations should assist the analyst in determining whether a user's behaviors are indicative of something concerning that should be investigated further or do they appear to be abnormal, but explainable in relation to the user's assigned tasks. Domain analysis tasks performed in support of explanation research entailed the development of analytic workflows using the PRODIGAL AI to perform mock-up investigations of Red Team user behaviors from various scenarios in the Vegas data. At each step of the mock-up investigations of Red Team users' behaviors, we described the type of information that an analyst would need to make a decision whether to investigate the user further for potential insider threat activities.

82	DT_FILE_EVENT_COUNT	
82_1	File_Event_Count_where_filename_extension_is_Text	Count of the number of file events where the extension is realted to a text application: .doc,.docx.,.log,.msg,.odt,.pages,.pages,.rtf,.tex,.txt,.wps,.wpd,.ttf,
82_2	File_Event_Count_where_filename_extension_is_Spreadsheet	Count of the number of file events where the extension is related to a spreadsheet application: .csv,.xls,.xlsx,.xlr
82_3	File_Event_Count_where_filename_extension_is_Presentation	Count of the number of file events where the extension is related to a presentation application: .ppt, .pptx,.pps
82_4	File_Event_Count_where_filename_extension_is_Data	Count of the number of file events where the extension is related to a data file: .dat,.sdf,.vcf,.ical,.log,.accdb,.db,.dbf,.mdb,.sql,.pst,.al,.dwg,.dxf,.kml,.kmz
82_5	File_Event_Count_where_filename_extension_is_Multimedia	Count of the number of file events where the extension is related to a multimedia application: .aif,.ai,.iff,.m3u,.m4a,.mp3,.mpa,.ra,.wav,.wma,.asx,.avi,.flv,.m4v,.mov,.mp4,.mpg,.rm,.swf,.vob,.wmv,. 3dm,.3ds,.max,.obj,.bmp,.gif,.jpg,.mpg,.png,.psd,.pspimage,.thm,.tif,.tiff,.ai,.dem,.gam,.nes,.rom,.sav,. 3g2,.3gp,.cdr,.torrent
82_6	File_Event_Count_where_filename_extension_is_Executable	Count of the number of file events where the extension is related to an Executable file: .apk,.app,.bat,.cgi,.com,.exe,.gadget,.jar,.pif,.vb,.wsf,.bim,.cmd,command,.cpl,.ins,.ipa,.isu,.job,osx,.out,.prg,.reg
82_7	File_Event_Count_where_filename_extension_is_SystemFiles	Count of the number of file events where the extension is related to a System application: .cab,.cpl,.cur,.deskthemepack,.dll,.dmp,.drv,.icns,.ico,.lnk,.sys,.cfg,.ini,.prf,.sys
82_8	File_Event_Count_where_filename_extension_is_CompressedFiles	Count of the number of file events where the extension is related to a file compression application: .7z,.cbr,.deb,.gz,.pkg,.rar,.rpm,.sitx,.tar.gz,.zip,.zipx
82_9	File_Event_Count_where_filename_extension_is_Code	Count of the number of file events where the extension is related to software development: .c,.class,.cpp,.cs,.dtd,.fla,.h,.java,.lua,.m,.pl,.py,.sh,.sin,.vcxproj,.xcodeproj,.swift
82_10	File_Event_Count_where_filename_extension_is_Backup	Count of the number of file events where the extension is related to file backups: .bak,.tmp
82_11	File_Event_Count_where_filename_extension_is_Internet	Count of the number of file events where the extension is related to the web: .asp,.aspx,.cer,.cfm,.csr,.css,.htm,.html,.js,.jsp,.php,.rss,.xhtml

Figure 17. *Example of the different finer-grained features we believe we could extract based on Feature 82.*

Analysis of results against the Red Team data sets and developing analytic workflows to support explanation research identified the potential of finer-grained features from existing data types to improve PRODIGAL's performance. Finer-grained features would use subsets of attributes within a data type to specify new features that focus on discrete types of user activities. For example, Feature 82 pertains to the count of file events. Analyzing Vegas data, Red Team scenarios, and PRODIGAL results, we determined that identifying specific types of files created would be useful in detecting malicious insider behavior, and that the Vegas file data attributes would support the specification of such a feature. **Figure 30** gives an example of the different finer-grained features we believe we could extract based on Feature 82, File Event Count.

We researched attributes from with all Vegas data types and designed over 300 potential new, finer-grained features that could be extracted and used as input to PRODIGAL anomaly and scenario-inspired detectors. We did not, however, specify new features in Phase 2 as we focused engineering tasks on improving system performance to support transition activities. Research on finer-grained features could be resumed in another, separate insider threat research with user activity monitoring data.

1.6. Content Analysis Research (Leidos SOW 1.3)

In the content analysis research, we tested the theory that terms, phrases, and concepts from instant messages (IM) in the Vegas data set would indicate users' job functions or project roles and whether these semantic indicators could enhance the detection of malicious insiders (i.e., Red Team users in Vegas data). We used IM data because we believe that users would be more

expressive and less guarded in their communication style than in corporate email. We developed a training corpus of 500 IMs from 20 user days from September and October 2012 Vegas data (with no Red Team inserts).. We divided the training set between IMs that contained negative and non-negative sentiments. When applied a naïve Bayes classifier to extract labels and developed feature counts of negative expressions, and then processed the data on Vegas data with Red Team inserts. We obtained accuracy of 70 percent in detecting negative sentiment using the negative sentiment classifiers developed through the training set. For the detection of job roles by concept, we focused on system administrators, software developers, test engineers, and managers. We identified over a dozen concepts that could fit in the context of a PRODIGAL scenario. Examples of concepts (and related scenarios) include: (1) organizational changes (IP Thief); (2) disgruntlement (IP Thief, Saboteur); (3) time wasting (Careless User); (4) negative sentiment directed toward other employees (Rager, Saboteur, IP Thief); and (5) job changes (IP Thief, Saboteur). We did not experiment with detecting job role based on topic in Phase 1 and did not perform additional content analysis research in Phase 2.

1.7. Initial Experiments with Surrogate Data (All SOW 1.3)

To jump start algorithm refinement and experimentation, the PRODIGAL team employed data from StackOverflow, a set of collaborative question-and-answer webs sites. We chose this data for use prior to SureView data with red team inserts becoming available. StackOverflow supports question, answer, and comment posts by a community of account-holders and provides several means of rewarding participation. While the site is well policed by its community, there are known schemes for gaming the rewards. StackOverflow published full data dumps every quarter for research in Social Network Analysis.

We inserted several target scenarios into the database of activity in the largest StackOverflow site (Programming). These scenarios simulated sabotage (Reputation Trashing), fraud (Reputation Gaming), and subterfuge (Cloning Accounts). This enhanced dataset allowed initial algorithm refinement over a common problem domain. Massive-scale graph analysis and community detection were run. For example 785K communities of voters were identified in 4 seconds. Interactive methods for graph exploration were refined, and were shown to enable an analyst to find networks of cloned accounts. Results from 5 anomaly detection algorithms were reported in October 2011 and several of these demonstrated enough promise to be continued into the SureView experiments. For example, Relational Pseudo Anomaly detection ranked all inserted instances in the top 1% of ranked accounts and Relational Anomaly Detection with Markov Algorithms found 67 of 103 inserted clones (see **Figure 18** and **Figure 19**).

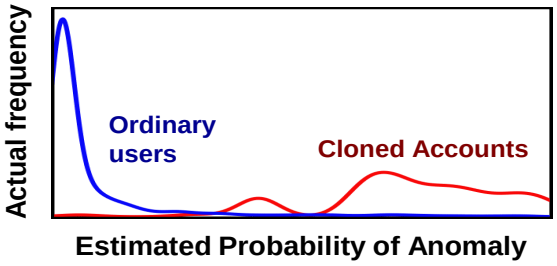


Figure 18. *Relational Pseudo-Anomaly Detection Likelihood Estimates – StackOverflow*

#	Reputation PM	Reputation DV	Badge	Post
1	FigBug	Col. Shrapnel	errx	SLaks
2	ScottE	Gold	Coding Kitten	Robert Harvey
3	David Cour.	There is noth.	penelope	Patrick H?se
		nder	Vlad Patryshev	doahou
				Michael Mrozek

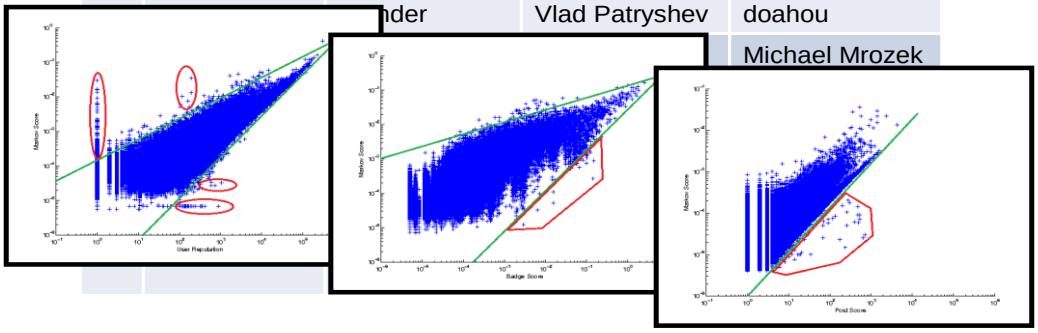


***Figure 19**. Relational Anomaly Detection with Markov Algorithms – StackOverflow Data.*

1.8. Experimentation with Live Data using the PRODIGAL Framework (All SOW 1.3; SOW 2.2)

The following sections will describe in detail the experimental setup and results with which we evaluated the performance of a large, diverse suite of anomaly detection methods. These methods included detectors based on the anomaly detection research previously discussed in this document as well as detectors designed to target specific scenarios and detectors based on statistical outlier feature values.

The primary focus of our efforts in Phase 1 was the establishment and operation of the PRODIGAL experimental framework, and beginning to evaluate detectors performance in isolation. With Phase 2, while continuing to experiment with detector performance in an increasingly diverse set of Red Team targets over accumulating data months, our research focus moved to three areas:

- Designing and experimenting with methods to combine the results of the detector suite (described in section 1.11);
- Developing and experimenting with methods by which the anomaly detection results could be explained to support both human and automated analysis (described in section 1.12); and
- Implementing an Analyst Interface (described in section 1.18) that could provide useful access and insight to a human counter-threat analyst of the data and threat estimates generated by our detectors.

1.8.1. Test Data and Red Team Scenarios

Test data for experimentation consists of a database of 5,500 users' computer usage. The data collection system, SureView (reference [SureView 2013]), records all user behaviors for specified activities, such as logon/logoff, email, file actions, instant message, printer, process, and URL events for a calendar month. On average, there are 1000 events per user per active day. Data are made available on a monthly basis. The data provider anonymizes all user identification

(ID) and other personally identifiable information (PII) in the data set and hashes all such information related to user events to randomly generated but internally consistent designators.

Separately from the data collection process, an independent Red Team developed scenarios reflecting their field experience of threat behaviors. ([Cappelli 2012], [Wallnau 2014]) Scenarios encapsulated specific insider threat actions that are superimposed on the actions of real users identified as appropriate to particular roles in the scenario. The Red Team inserted up to five instances of scenarios (with variations) in a data month for a total of 54 instances of 28 distinct scenarios. [Wallnau 2014] **Table 2** contains brief synopses of the inserted Red Team scenarios

The Red Team, in an effort to avoid evaluation bias, designed its scenarios independently of our detection methods. Likewise, we did not review scenario specifics nor train our detectors on the test data to avoid over-fitting. Scenario descriptions in **Table 2** were provided after the fact for purposes of evaluating our methods; the Red Team continually added new scenarios as the research was ongoing. Neither the Red Team nor our team claims that the set of scenarios included in **Table 2** is complete.

Table 2. *Brief synopses of the inserted Red Team scenarios*

Scenario Name (No. of Instances)	*Scenario Synopsis*
Anomalous Encryption (2)	An insider passes proprietary information to an outsider secretly encrypting files and emailing them from his work email to his personal email.
Blinded Me With Science (1)	(BENIGN) Subject is constantly trying to learn new things and often spends free time reading interesting research; uses Google Scholar and other tools. Subject downloads the article/document/file workstation; sends the document to a personal webmail address so that it can be reviewed while away from work.
Bollywood Breakdown (1)	Subject is convinced by Indian official posing as a hospital administrator to exfiltrated IP and classified documents in exchange for preferential medical treatment for a family member.
Bona Fides (2)	Espionage volunteer prints a bona fides package and takes it to a foreign embassy.
Breaking the Stovepipe (3)	Subject accidentally exposes his access to proprietary information of company A to company B. Company B's contact bribes subject to share the information.
Byte Me Middleman (2)	Variation of Byte Me where all illicit electronic communication with Subject goes through CCA. Other co-conspirators communicate with Subject in person (so are unobserved).
Byte Me (2)	Subject uses special access to badge entry system to profit by selling special access badges to smokers to be used at a more convenient unauthorized doorway. Subject regularly sweeps the database to remove these extra entries.
Circumventing Sureview (20)	A user circumvents SureView monitoring to commit a crime.
Conspiracy Theory (2)	Subject has been a model employee until very recently, when Subject stopped taking medications for treatment of paranoid schizophrenia. Over a period of several weeks, Subject begins displaying unusual behavior by claiming that the organization is having Subject followed outside of work. Subject shares his/her concerns with co-workers who ask if Subject has proof that this is happening. Subject claims that Subject's followers are "too smart for that" and that they always stay in parking lots "where they know there are no cameras that can record them." Subject writes down license plate numbers and descriptions of the followers "only to have them stolen" out of Subject's wallet/purse. Subject's paranoia becomes so acute that immediately following the end of a work day, Subject emails a supervisor stating simply "I quit! So stop having me followed or I will be forced to defend myself!!" One to two hours later Subject remotely logs back into the company network through the VPN and deletes vast amounts of files from the shared network directories.

Credit Czech (1)	Subject runs an illicit business trafficking in stolen credit card numbers using the organization's IT resources. He acts as a middleman between various external purveyors of stolen numbers and a Russian operative who buys the collected numbers.
Czech Mate (1)	Similar to Credit Czech but the new protocol calls for twice-daily emails to Subject's Russian counterpart in order to keep the operation alive.
Exfiltration of Sensitive Data Using Screenshots (3)	An employee steals proprietary/sensitive docs by taking screenshots of specific pages recursively encrypting the files and emailing them to a webmail address.
Exfiltration Prior to Termination (2)	An employee is leaving the company and decides to take all of their emails and files with them.
From Belarus With Love (2)	A user circumvents SureView monitoring to commit a crime.
Gift Card Bonanza (1)	Under financial pressure due to sister's medical expenses subject uses co-worker's email addresses to enter paying surveys and enlists help of email administrator to hide the mis-appropriation.
Hiding Undue Affluence (2)	An employee possesses undue affluence because of ongoing espionage activity. They need to hide the existence of the money from investigators and they perform research on how to do so.
Indecent RFP (3)	Subject uses an inappropriate relationship with another employee to illegally influence vendor selection for a lucrative catering contract in order to obtain personal financial gain.
Insider Startup (7)	Three co-conspirators collude to steal company IP. They coordinate the synchronized theft of proprietary information before leaving the company.
Job Hunter (1)	Subject is having trouble getting through the counterintelligence scope polygraph (CSP) required for Subject's job position. Subject decides to find employment with a foreign company so that the CSP will not be required. Subject reaches out to a contact to ask if he or his sister have any contacts in the Israeli aerospace company "ISI." Contact's sister instructs Subject to contact a trade representative in the Israeli consulate. Subject follows up on the lead and ends up sharing sensitive/classified information with the consular in order to secure the new job with ISI.
Layoff Logic Bomb (2)	An insider worried about layoffs uploads a logic bomb into the IT system that will detonate unless he disables it.
Manning Up (2)	An insider emulates Bradley Manning and researching similar techniques while at work.
Manning Up Redux (1)	An insider emulates Bradley Manning and researches detection counter-measures and scripts code that will upload large amounts of files through a custom DNS.
Masquerading 2 (2)	Subject sets up a rogue SSH server on another user's machine. They also make a copy of the local Windows password file and copy the file off over the network.
Passed Over (2)	Subject learns that his/her project is being phased out in a re-org becomes extremely disgruntled makes demands and threats to his/her leadership and then installs malware on several machines before submitting a resignation.
Naughty by Proxy (4)	A disgruntled employee seeks revenge by logging on to her manager's computer and visiting questionable websites.
Outsourcer's Apprentice (3)	A software developer outsources his job to China and spends his workdays surfing the web. Some surfing activity occurs on his main workstation while the subcontractor is active, but most of it occurs on a second laptop he uses to try to minimize his interference with the subcontractor. He pays just a small fraction of his salary to a company based in China to do his job. The developer provides remote access to his machine by providing his VPN credentials to the Chinese company and enabling Terminal Services on his workstation. The Chinese consulting firm sends the developer PayPal invoices for the work performed, and the developer pays them.

Parting Shot (1)	Subject was the first one into the office one morning and found a document on a printer that appeared to be a list of 12 people scheduled to be terminated in the next few months. The list was marked as "company most private" and stated that the terminations were in order to achieve company cost-reduction targets and that individuals would not be notified until their last day of work. The document provided no indication of severance pay or other monetary compensation being provided to the terminated individuals. Subject, as well as some friends of Subject, were named in the list of expected terminations. Subject then posts the information to the internal time sheet website and tells some co-workers to look there.
Parting Shot 2 - Deadly Aim (1)	Deadly Aim: Subject is disgruntled after finding out that Subject is on a list of employees that will be terminated without warning at an unknown point in the future. Subject previously conspired with CCA to make the whole list public to all employees. Subsequently, Subject found employment with another company and decided, as a parting shot, to embarrass the company's senior executives following Subject's resignation. Subject obtains a list of salaries for the senior executives (Presidents, C-Level staff, etc.) and posts the information where it can be accessed by all employees.
Passed Over (4)	Subject learns that his/her project is being phased out in a company re-organization. Subject becomes extremely disgruntled, makes demands and threats to his/her leadership, and then installs malware on several machines before submitting a resignation.
Panic Attack (2)	Subject was 'recruited in place' by a FIS in order to provide highly specialized U.S.-developed zero-day exploit code that could be usable by FIS in offensive operations. Subject has displayed "low and slow" operational activity in order to evade detection by U.S. Counterintelligence. That is to say, Subject displays highly specialized tradecraft not normally observable by common collectors. For example, Subject is only "operational" on an aperiodic schedule and does not exfiltrate proprietary information through email, USB, printing, or other computer-observable means. However, Subject is informed by a company security representative that Subject's position has been newly designated by the government customer as requiring a "Counterintelligence Scope Polygraph" (CSP). Subject does some research to determine what a CSP (a.k.a. CI Poly) covers and tries to discern countermeasures that would defeat the CSP, but ultimately decides it would be too risky and resigns in order to avoid detection. In his/her panic, Subject accidentally exfiltrates a large number of documents before resigning.
Selling Login Credentials (1)	One insider who has system privileges sells (with the unwitting help of two other insiders) dummy accounts and temporary access to outsiders.
Snowed In (4)	A sys-admin discovers and reports a website security vulnerability but is reprimanded for it. He then discovers a classified document about a secret government surveillance program and carefully exfiltrates the document along with legitimate duties.
Strategic Tee Time (1)	(BENIGN) A company executive has requested a two-day offsite strategic planning meeting with three other Directors or VPs. During the planning process, one of the participants recommends a golf outing directly following the offsite. The various admins have to exchange information concerning the agenda, hotel arrangements, and golf outing. Email attachments of agendas and itineraries are exchanged.
Survivor's Burden (3)	The subject is disgruntled after his team experienced layoffs (and a logic bomb), greatly increasing his workload. He hopes to become the team lead, but is passed over for the position and takes matters into his own hands by stealing company IP using DropBox.
The Big Goodbye (1)	(BENIGN) Admin is coordinating a farewell luncheon for an individual leaving the company (for retirement or other reasons). Admin wants to ensure that people submit meal choices for the luncheon, sign a farewell card and contribute toward a farewell gift should they desire.

What's the Big Deal (1)	Subject has been storing data on a cloud storage site so that she can work while at home and on vacation. Subject considers herself underpaid and is actively seeking employment with competitors. Subject learns that the cloud company she has been using has been hacked. Believing nothing bad is likely to happen, Subject decides not to report the fact that the company's documents may have been compromised. Subject recognizes that she should delete the files sent to the third party server, but decides to keep a good number that she produced. She deletes only some of the more sensitive files.

1.8.2. A Diverse Suite of Anomaly Detectors

Our approach built on the techniques described in [Senator 2013], [Young 2013] and [Young 2014]. Nearly all detectors in PRODIGAL operate on vectors of domain-based features, each derived from a defined set of entities and a fixed period (extent). Most experimentation in PRODIGAL was performed on user-days as a convenient and reasonable point in the vast set of possible targets for anomaly detection. **Table 3** lists the features implemented in the experiments described below.

Table 3 *Features implemented in the experiments.*

URL_EVENT_COUNT	FILE_CD_DRIVE_EVENT_COUNT
URL_UPLOAD_COUNT	FILE_DISTINCT_REMOVABLE_DRIVE_COUNT
URL_DOWNLOAD_COUNT	FILE_COPIES_TO_REMOVABLE_COUNT
URL_DISTINCT_URL_COUNT	FILE_COPIES_FROM_REMOVABLE_COUNT
URL_DISTINCT_DOMAIN_COUNT	FILE_CREATION_COUNT
URL_DISTINCT_WORKSTATION_COUNT	FILE_FOLDER_CREATION_COUNT
URL_DOMAINS_PER_USER_COUNT	FILE_DISTINCT_FILES_COUNT
URL_FIRST_EVENT_TIME	FILE_DISTINCT_FILES_ON_REMOVABLE_COUNT
URL_LAST_EVENT_TIME	FILE_DISTINCT_WORKSTATION_COUNT
URL_UPLOAD_TO_ANOMALOUS_COUNT	FILE_FIRST_EVENT_TIME
URL_DOWNLOAD_TO_ANOMALOUS_COUNT	FILE_LAST_EVENT_TIME
LOGON_EVENT_COUNT	FILE_FIRST_REMOVABLE_EVENT_TIME
LOGON_DISTINCT_WORKSTATION_COUNT	FILE_LAST_REMOVABLE_EVENT_TIME
LOGON_FIRST_EVENT_TIME	FILE_SRC_FIXED_DRIVE_EVENT_COUNT
LOGON_LAST_EVENT_TIME	FILE_SRC_REMOVABLE_DRIVE_EVENT_COUNT
PRINTER_PAGES_PRINTED_COUNT	FILE_SRC_NETWORK_DRIVE_EVENT_COUNT
PRINTER_JOBS_SUBMITTED_COUNT	FILE_SRC_CD_DRIVE_EVENT_COUNT
PRINTER_DISTINCT_COUNT	FILE_DEST_FIXED_DRIVE_EVENT_COUNT
PRINTER_DISTINCT_WORKSTATION_COUNT	FILE_DEST_REMOVABLE_DRIVE_EVENT_COUNT
PRINTER_PER_USER_COUNT	FILE_DEST_NETWORK_DRIVE_EVENT_COUNT
PRINTER_FIRST_EVENT_TIME	FILE_DEST_CD_DRIVE_EVENT_COUNT
PRINTER_LAST_EVENT_TIME	FILE_FIX_EVENTS_VS_FILE_EVENTS
DEVICE_EVENT_COUNT	FILE_REMOVABLE_EVENTS_VS_FILE_EVENTS
DEVICE_DISTINT_WORKSTATION_COUNT	FILE_NETWOERK_EVENTS_VS_FILE_EVENTS
DEVICE_DISTINT_DEVICE_COUNT	FILE_CD_EVENTS_VS_FILE_EVENTS
DEVICE_FIRST_EVENT_TIME	FILE_FIX_DRIVE_VS_REMOVEABLE_DRIVE
DEVICE_LAST_EVENT_TIME	FILE_FIX_DRIVE_VS_NETWORK_DRIVE
EMAIL_EVENT_COUNT	FILE_FIX_DRIVE_VS_CD_DRIVE
EMAIL_RCVD_VIEWED_COUNT	FILE_DISTINCT_VS_DISTINCT_ON_REMOVE
EMAIL_RCVD_DISTINCT_COUNT	FILE_DISTINCT_WS_VS_FIXED_DRIVE_EVENTS
EMAIL_RCVD_DISTINCT_SENDER_COUNT	URL_DISTINT_VS_DISTINT_DOMAIN
EMAIL_RCVD_DISTINCT_WRKST_COUNT	URL_UPLOADS_VS_DOWNLOADS

EMAIL_RCVD_ATTACHMENT_COUNT	URL_UPLOADS_VS_DISTINCT_WS
EMAIL_SENT_COUNT	URL_EVENTS_VS_UPLOADS
EMAIL_SENT_DISTINCT_RECIPIENT_COUNT	URL_EVENTS_VS_DOWNLOADS
EMAIL_SENT_TO_COUNT	URL_DISTINCT_URLS_VS_UPLOADS
EMAIL_SENT_CC_COUNT	URL_DISTINCT_URLS_VS_DOWNLOADS
EMAIL_SENT_BCC_COUNT	URL_DISTINCT_DOMAINS_VS_UPLOADS
EMAIL_SENT_DISTINCT_WORKSTATION_COUNT	URL_DISTINCT_DOMAINS_VS_DOWNLOADS
EMAIL_SENT_ATTACHMENT_COUNT	URL_DISTINCT_WS_VS_EVENTS
EMAIL_SENT_AVERAGE_RECIPIENT_COUNT	URL_DOWNLOADS_VS_DISTINCT_WS
EMAIL_SENT_FISRT_EVENT_TIME	DISTINCT_REMOVABLE_VS_URLS_EVENTS
EMAIL_SENT_LAST_EVENT_TIME	DISTINCT_REMOVABLE_VS_URLS_UPLOADS
EMAIL_RCVD_FROM_ANOMALOUS_SENDER_COUNT	DISTINCT_REMOVABLE_VS_URLS_DOWNLOADS
EMAIL_SENT_TO_ANOMALOUS_ADDR_COUNT	REMOVABLE_FILE_EVENTS_VS_URL_EVENTS
FILE_EVENT_COUNT	FIXED_FILE_EVENTS_VS_URL_UPLOADS
FILE_FIXED_DRIVE_EVENT_COUNT	FIXED_FILE_EVENTS_VS_URL_DOWNLOADS
FILE_REMOVABLE_DRIVE_EVENT_COUNT	RATIO_209_VS_RATIO_233
FILE_NETWORK_DRIVE_EVENT_COUNT	EMAIL_SENT_VS_RCVD_RATIO

PRODIGAL employs a large number of diverse detectors of three types: (I) indicator-based, (A) anomaly-based, and (S) scenario-based. **Table 3** lists the detectors configured in PRODIGAL.

Indicator-based detectors use statistical outlier techniques over sub-sets of features related to one or two particular types of activity such as file or web access. PRODIGAL computes a statistical outlier score for each value by comparing it against all other users for that date using the cumulative distribution function (CDF) of the logistic distribution with mean and variance of this population. This score is normalized to [0; 1] and is easily compared with other features' scores. This score can be viewed as a "marginal" explanation of the anomaly of the user-day, because it estimates the likelihood that the feature value is greater than those of other users from the base population. An explanation of a scored entity is a list of features plus outlier scores.

Anomaly detectors employ complex models that focus on different aspects of the data, e.g. structural features, semantic features, or temporal features, and then search through the entire feature space to identify potential anomalies. Features typically consist of observed actions, aggregates, or ratios, such as URLs accessed by a user, the number of print jobs by a user, or the ratio of the number of files copied to removable media compared to the total number of files actions. Relational features such as the email and text-message communication graphs are used to provide comparison groups in some detectors. Different approaches to feature normalization are incorporated into variants of the same detection models used in PRODIGAL [Senator 2013].

Scenario-based detectors are inspired by the scenarios described in [Cappelli 2012], but are developed independently of the Red Team scenario descriptions and inserts. They consist of a combination of indicator-based and anomaly-based detectors and classifiers in a specified workflow, structured to reflect a hypothesized combination of real world actions that are likely to discriminate between the scenario of interest and other, mostly legitimate, actions. Six scenario-based detectors have been deployed in varying stages of development in PRODIGAL. They focus on particular sub-spaces of features that are relevant to a particular scenario, as well as subsets of target users and/or time periods, as suggested by the scenario. In this way, domain knowledge of both activity type and relevant comparison peer groups are incorporated into the detection.

A particular detector specification incorporates, in addition to the algorithm, a set of features, a baseline population for comparison (i.e., a peer group), a time period for the baseline activity, time granularity for potential detection, a particular approach to feature normalization, and other relevant aspects. Baselines for comparison may be cross-sectional (i.e., compare a users actions over a particular time period with that of other users in a peer group over some time comparable time period) or temporal (i.e., compare a user with his/her own behavior over different time periods), or both.

We develop their specifications using the Anomaly Detection Language (ADL) that was introduced in [Senator 2013] and [Young 2013]. We have found specification of a complex planned detection in ADL to be extremely useful in the process of developing and testing combinations of feature sub-setting, classification, anomaly detection, and threat ranking mechanisms. We now describe the scenarios on which our scenario-based detectors were based and map each to RT scenarios to which we believe they best correspond.

Saboteur: An insider uses corporate information technology (IT) resources to harm an organization or an individual. Saboteurs are technical, such as a system administrator, and have privileged access to systems. The saboteur plans his attack before leaving the organization and executes the attack as he leaves. Corresponding RT Scenario(s): Gift Card Bonanza, Layoff Logic Bomb, Passed Over.

IP Thief : An insider uses corporate IT resources to steal IP. IP thieves are generally scientists, engineers, or salespeople, they generally steal what they consider to be their own work for their own private gain. Corresponding RT Scenario(s): Anomalous Encryption, Bona Fides, Bollywood Breakdown, Breaking the Stovepipe, Exfiltration Prior to Termination, Exfiltration of Sensitive Data Using Screenshots, Manning Up, Manning Up (Redux), Snowed In.

Fraudster: An insider uses IT for destroying, denying, or degrading an organization's information or systems for personal gain or to commit a crime. Fraudsters are lower-level employees, are often motivated by financial need - hardship, greed, etc. Sometimes they are recruited by outsiders in collusion with other insiders. Corresponding RT Scenario(s): Byte Me, Credit Czech, Czech Mate, Masquerading 2.

Ambitious Leader : The Ambitious Leader is an IP thief who is motivated by ambition to steal as much as possible before leaving the organization. To do so, he recruits other insiders to get access to all parts of the IP being stolen. Corresponding RT Scenario(s): Insider Startup, Selling Login Credentials.

Careless User: The insider is not intentionally malicious but, through blatant disregard of corporate IT policies, exposes the group to a comparable level of risk similar to the Saboteur scenario. Corresponding RT Scenario(s): None.

Rager: The insider has outbursts of strong, vociferous, abusive, and threatening language in Email/Webmail/IM toward other insiders or against the organization in general. These outbursts coincide with anomalies in other data types, e.g., Logons, URL, indicating a potential fundamental change in behavior. Corresponding RT Scenario(s): None.

Note that there are no Red Team scenarios corresponding to some of our detectors. Since we did not know what scenarios the Red Team might choose, we constructed a broad set of detectors to cover many envisioned scenarios. A reverse mapping from Red Team inserts to detector scenarios reveals examples of this: The Bona Fides datasets included espionage activities we had

not considered at the time; and the same is true for the system manipulations done by users in the Masquerading 2 and Circumventing SureView datasets.

Table 4. *User-Day Detectors in PRODIGAL*

Algorithm	Type	Description
TBAD	A	Temporal Based Anomaly Detection
VSM	A	Vector Space Models (User Day)
GFADD:92-95-0	I	Grid-based Fast Anomaly Detection with Duplicates (GFADD) (File creation vs. distinct file no grid)
GFADD:95-96-8	I	GFADD (Distinct file count vs. files on remov. count grid:8)
GFADD:89-90-0	I	GFADD (Copies to vs. from removable drives no grid)
GFADD:89-90-8	I	GFADD (Copies to vs. from removable drives grid:8)
GFADD:82-84-8	I	GFADD (File vs. removable drive events grid:8)
GFADD:83-85-0	I	GFADD (Fixed event count vs. network event count no grid)
GFADD:83-85-8	I	GFADD (Fixed event count vs. network event count grid:8)
GFADD:83-84-8	I	GFADD (Fixed vs. removable event counts grid:8)
GFADD:85-88-0	I	GFADD (Network events vs. distinct removable drives no grid)
GFADD:84-88-0	I	GFADD (Removable drive events vs. distinct no grid)
GFADD:84-88-8	I	GFADD (Removable drive events vs. distinct grid:8)
GFADD:84-85-0	I	GFADD (Removable drive events vs. network events no grid)
GFADD:84-85-8	I	GFADD (Removable drive events vs. network events grid:8)
GMM:RD	A	Gaussian Mixture Model (Raw count features user-day scores)
GMM:QD	A	Gaussian Mixture Model (Quantile features user-day scores)
EGMM:RD	A	Ensemble Gaussian Mixture Model (Raw count features user-day scores)
EGMM:QD	A	Ensemble Gaussian Mixture Model (Quantile features user-day scores)
CROSS:RD	A	Cross Prediction (Raw count features userday scores)
CROSS:(QD	A	Cross Prediction (Quantile features userday scores)
RIDE:RD	A	Repeated Impossible Discrimination Ensemble (Raw count features user-day scores)
RIDE:QD	A	Repeated Impossible Discrimination Ensemble (Quantile features user-day scores)
IFOR:RD	A	Isolation Forest (Raw count features userday scores)
IFOR:QD	A	Isolation Forest (Quantile features userday scores)
CADE:R	A	Classifier-Adjusted Density Estimation (Raw features)
CADE:UP	A	Classifier-Adjusted Density Estimation (UP features)
PDE:R10K	A	Pseudo-likelihood Density Estimator (raw features 10k training)
PDE:UP10K	A	Pseudo-likelihood Density Estimator (UP features 10k training)
PDE:UP	A	Pseudo-likelihood Density Estimator (UP feature set)
Saboteur	S	Scenario: Saboteur (Variant 2)
IP Thief	S	Scenario: IP Thief (Variant 1)
Fraudster	S	Scenario: Fraudster (Variant 1)
Amb. Lead.	S	Scenario: IP Thief Ambitious Leader (Variant 1)
File	I	Indicator: File Activity (Variant 1)
URL	I	Indicator: URL Activity (Variant 1)
File-URL	I	Indicator: File vs. URL (Variant 1)
URL,File-Log.	I	Indicator: URL and File vs. Logon
Careless	S	Scenario: Carless User
Rager	S	Scenario: Rager

1.8.3. Measuring Detector Performance

By August 2012, with stability of the SureView database accomplished and the start of monthly insertion of target scenario instances by the CERT red team, we were able to apply the PRODIGAL Framework to ingest and transform data and to run the entire set of algorithms for graph analysis and community, outlier, and anomaly detection. This allowed us to explore combinations of features, entities, baselines and peer groups, and detection methods for Insider

Threat Detection. Our primary focus in this phase was on methods to detect starting points for further analysis. We used each available month of data independently. Each month had an unknown number of independent red team inserts.

Starting in August 2012, we ran more than 200 detection jobs each month using statistical outliers of single features, anomaly of activity indicator feature vectors, a variety of complex, inter-activity anomaly models, and scenario-based models involving all of these. A total of 1839 runs produced rankings of observed entities and spans variation over background, non-target activity as well as target scenario. We applied several performance metrics to understand the effectiveness and robustness of each method, and are beginning to explore the potential for combination of methods.

Table 5. *Two months of detection focused on a single scenario.*

Month	Algo	Detection Method	5(0)	10(0)	50(0)	100(0)	500(0)	AUC	AvgLift
Sept	UMASS-1	RPAD up feature normalization	2	2	3	11	72	0.970	17.42
Sept	UMASS-1	RPAD dp feature normalization	2	4	20	26	57	0.863	24.07
Sept	UMASS-1	RPAD raw feature set; naive bayes; uniform pseudo-anomaly	0	0	10	26	56	0.879	16.06
Sept	SAIC-6	Indicator Anomaly Detection - File	0	1	17	33	54	0.881	10.58
Sept	SAIC-3	Scenario - IP Thief	0	0	7	16	54	0.851	9.79
Sept	SAIC-8	Indicator Anomaly Detection - File vs URL	1	2	4	9	50	0.732	6.04
Sept	SAIC-5	Scenario - Ambitious Leader	9	12	43	46	48	0.806	34.05
Sept	UMASS-2	RDE alpha version; raw feature set; 10k training	0	0	7	12	42	0.864	10.75
Oct	UMASS-1	RPAD up feature normalization	2	2	5	11	37	0.979	30.33
Oct	SAIC-6	Indicator Anomaly Detection - File	0	0	2	14	31	0.874	8.42
Oct	UMASS-1	RPAD g129dm feature normalization	0	0	1	3	29	0.914	13.70
Oct	SAIC-8	Indicator Anomaly Detection - File vs URL	0	0	2	8	28	0.824	6.02
Oct	UMASS-1	RPAD raw feature set; naive bayes; uniform pseudo-anomaly	0	0	0	3	20	0.909	9.17
Sept	SAIC-2	Scenario - Saboteur	0	1	4	6	20	0.746	3.79
Oct	SAIC-3	Scenario - IP Thief	0	0	0	3	15	0.839	7.34
Oct	SAIC-2	Scenario - Saboteur	0	0	0	0	15	0.810	3.07
Oct	SAIC-5	Scenario - Ambitious Leader	6	7	12	12	15	0.789	80.20
Sept	SAIC-1	Max(Cross & Long Outliers)	0	0	0	1	14	0.846	3.99
Sept	OSU-3	Ensemble GMM Density Estimation, Raw Counts	0	0	0	0	12	0.970	26.17
Oct	GTRI-5	Temporal Based Anomaly Detection	0	0	0	0	12	0.849	6.14
Sept	OSU-1	GMM Density Estimation using Raw Counts	0	0	0	0	10	0.940	7.83
Sept	OSU-4	RIDE via unusualness of counts vs. company	0	0	0	2	10	0.920	8.05
Sept	SAIC-4	Scenario - Fraudster	0	0	0	1	10	0.693	1.62
Sept	SAIC-9	Indicator Anomaly Detection - File vs URL vs Logon	0	0	3	4	8	0.530	1.26
Sept	OSU-2	Cross Prediction via unusualness of counts, vs company	0	1	1	1	7	0.872	8.86
Oct	SAIC-4	Scenario - Fraudster	0	1	1	1	7	0.713	4.57
Oct	OSU-4	RIDE via unusualness of counts vs. company	0	0	1	3	6	0.981	26.18
Oct	OSU-3	Ensemble GMM Density Estimation, Raw Counts	0	0	0	0	6	0.970	15.84
Sept	OSU-4	RIDE using Raw Counts	0	0	0	2	6	0.892	7.09
Oct	UMASS-2	RDE alpha version; raw feature set; 10k training	0	0	0	0	5	0.895	6.10
Sept	SAIC-7	Indicator Anomaly Detection - URL	0	0	0	1	5	0.477	0.91
Sept	CMU-6	Grid-based Anomaly Detection given Duplicates	2	5	5	5	5	0.301	2.19
Sept	GTRI-5	Temporal Based Anomaly Detection	0	0	0	0	3	0.502	1.00
Oct	CMU-6	Grid-based Anomaly Detection given Duplicates	1	1	1	2	3	0.465	1.77
Oct	OSU-3	Ensemble GMM via unusualness of counts, vs company	0	0	0	0	2	0.906	5.32
Oct	OSU-4	RIDE using Raw Counts	0	0	0	0	2	0.888	4.69
Oct	GTRI-4	Vector Space Models	0	0	1	2	2	0.694	8.64
Sept	GTRI-4	Vector Space Models	0	0	1	1	2	0.618	2.61
Oct	SAIC-9	Indicator Anomaly Detection - File vs URL vs Logon	0	0	0	0	2	0.425	0.87
Oct	OSU-1	GMM Density Estimation via unusualness of counts, vs company	0	0	0	0	1	0.881	4.18
Oct	OSU-2	Cross Prediction via unusualness of counts, vs company	0	0	0	0	1	0.833	3.15
Sept	OSU-3	Ensemble GMM via unusualness of counts, vs company	0	0	0	0	1	0.787	2.20
Sept	OSU-1	GMM Density Estimation via unusualness of counts, vs company	0	0	0	0	1	0.780	2.16
Oct	OSU-1	GMM Density Estimation using Raw Counts	0	0	0	0	0	0.900	4.99
Oct	SAIC-1	Max(Cross & Long Outliers)	0	0	0	0	0	0.828	3.27
Oct	SAIC-7	Indicator Anomaly Detection - URL	0	0	0	0	0	0.507	0.93
Oct	OSU-2	Cross Prediction using Raw Counts	0	0	0	0	0	0.388	0.92
Sept	OSU-2	Cross Prediction using Raw Counts	0	0	0	0	0	0.287	0.66

Experiments Focused on a Single Scenario

For the data months of September and October in 2012, the red team inserted instances primarily of one scenario - three insiders who collude over instant messaging and corporate email to steal IP and form a new company. While the particular scenario was unknown to the detectors and to the research team prior to the experiments, it comprises several types of activity we have identified as important indicators of insider threat and is similar to the IP Thief Ambitious Leader scenario without the presence of an identifiable leader. We discovered that the red team had inserted two variants of this scenario over the months of September and October 2012, inserting a total of six instances. A single instance of a second scenario that simulated users' circumventing SureView's data collection was also inserted in September.

We used this opportunity to study both effectiveness and robustness, and have analyzed the results of 484 detection runs over these two months. Each month comprised approximately 5000 active users with an average of 1000 evidence observations each user-day. September's red team targets consisted of 13 users with activity on 98 separate user-days and October had 6 target users with activity on 44 user-days. **Table 5** shows several of our performance metrics, AUC, Average Lift, and # of target user-days at several cut-off ranks for each detector operating on these months' data.

In the following sections, we will describe the results of continued experimentation with our suite of detectors over a period of 2 data years, and discuss a few examples of individual detection runs, which illustrate important findings.

Measuring Overall Detector Performance

The section above describes the data environment in which our prototype operates. Instances of Red Team scenarios are limited to one month duration and inserted as targets each calendar month. (CERT has found that 2/3 of known insider threat scenarios evolve over less than one month.) This allows for consistent, independent experiments.

The experiments reported here measure detection performance on user-day entity extents, a data structure derived from the collection of activities of one user over one day. (We limit entity extents to this size for these experiments, although PRODIGAL is capable of representing many others.) We consider a hit to be the ranking of a user-day above some threshold, and can measure the hit/false-alarm tradeoffs using measures of the rate of true positives and false negatives.

Addressing the Needs of Insider Threat Surveillance

Metrics were chosen to measure both detection accuracy of the individual algorithms and their contribution to the overall task of providing leads to an analyst. For the former, we compute the Receiver Operator Characteristic (ROC) curve and area under the curve (AUC) as well as the Approximate Lift Curve and Average Lift. AUC estimates discrimination, or the probability that a randomly chosen positive entity extent will be ranked higher than a randomly chosen negative one.

The choice of metric is critical to achieve the goal of providing workable leads to an analyst. Initial development has relied on the AUC metric, and that is what is reported here. However, that metric may be less suitable for a highly asymmetrical detection situation, where a very few positives must be identified high in the ranking to support an effective layered detection process in which analysts receive at least one lead from each (or most) scenarios near the top of the list and can then "connect the dots" to the rest of the malicious behavior.

Lift metrics, such as Average Lift, estimate the improvement in target density delivered to later stages of a multi-stage detection process. We also compute the number of positive hits ranked in the top k scored entity extents (for k = 5, 10, 50, 100, 500, etc.) and in the top p% of all scored entity extents (for p = .01, .05, .1, .5, 1, 5, etc.). The latter allow us to estimate anticipated detection success for fixed analyst workloads.

We investigated a number of lift-focused approaches, aiming to improve the detection of leads to better support a layered detection process. **Figure 20.** Feasibility of Insider Threat Detection based on Anomaly Detection depicts how such processes, individually with very modest lift values, can be combined to solve the problem. The unsupervised anomaly detection and ensemble methods we describe subsequently in this paper can achieve the indicated performance - in 85% of inserted instances, the lift of the highest ranked target user-day is better than 10. Hence, a layered approach, which includes analytics to follow the leads provided by PRODIGAL, could provide acceptable performance.

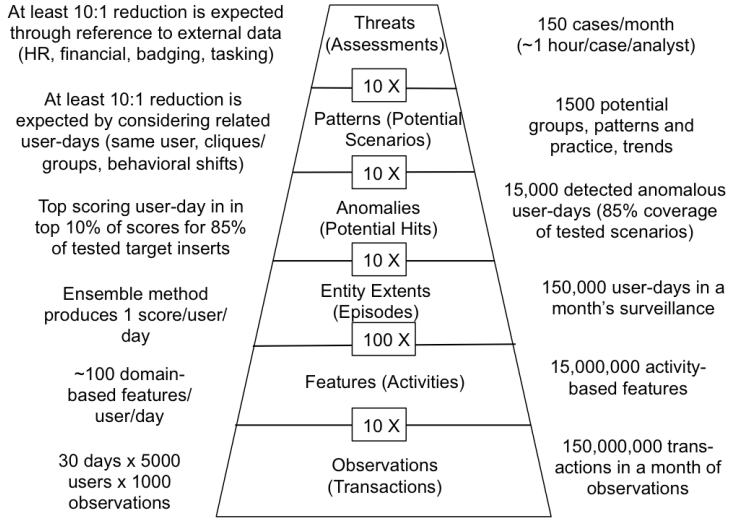


Figure 20. *Feasibility of Insider Threat Detection based on Anomaly Detection*

The following sections report on our investigation of unsupervised anomaly detection, ensemble methods for combining diverse detection results, and the role that domain-specific detectors may or may not play in the acquisition of leads of insider threat surveillance.

1.9. Experiments with Detector Diversity

An example of the experiments, which the PRODIGAL framework enabled, was experiments in Detector Diversity. The key idea in these experiments was to apply the "Divide and Conquer" approach to statistical anomaly detection by running over **meaningful** sub-populations of entity-extents. Then results would be combined as a detector ensemble. We noted that targeted user behavior is intermittent, interspersed with normal activities, and that aggregation of activities into features can "blur" significant values (aggregation over multiple generating processes decrease variance). Also, finding anomalies may depend on context – e.g. time of day, day of week, comparison peer group. Different malicious scenarios may appear as atypical behavior when viewed in certain contexts. Finally, ensemble combination methods such as the method we use in PRODIGAL rely on having a diverse set of detectors scoring the same entity-extents.

Since many anomaly detection algorithms incorporate ensembles of statistical models, essentially performing a divide and conquer partitioning of the feature space, we wanted to divide the space along semantically meaningful lines – work vs. non-work days, email communities, division function, etc.

The ensemble combination approach we are using relies on diversity among the input detectors for its power. Our hypothesis was that adding many variants of the same algorithms, run over diverse partitions of the data, could improve overall performance as compared to a single run of any detector or set of detectors.

We divided the existing feature space along meaningful lines. Entities were divided into peer groups as found by shared resources (see **Figure 34**) or by external information (**Figure 35**). Extents were divided by duration (day, week, partial day) and cycle (workdays, Sat/Sun/Holidays, all days).

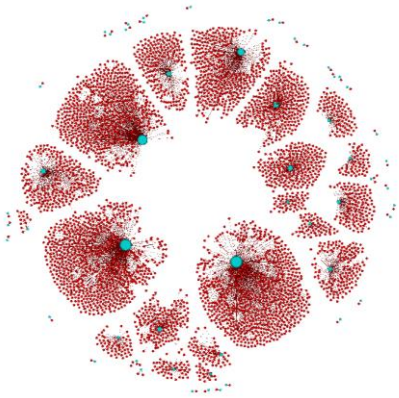


Figure 21. *Email communities (per GTRI Community Detection)*

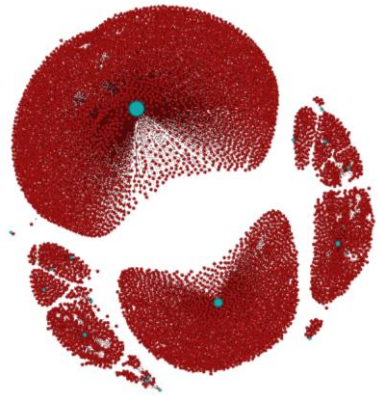


Figure 22. *Division function (per LDAP data)*

Preliminary results were promising. Using only full day extents, three statistical anomaly detector models, and varying by cycle and peer group, generated 27 variant detectors. The ensemble results of this set increased detection of one target scenario strongly, while weakly reducing another. ROC curves in **Figure 23** below show the results of each of the 3 detectors running on un-divided data and by comparison, the results of the ensemble of the 27 variants.

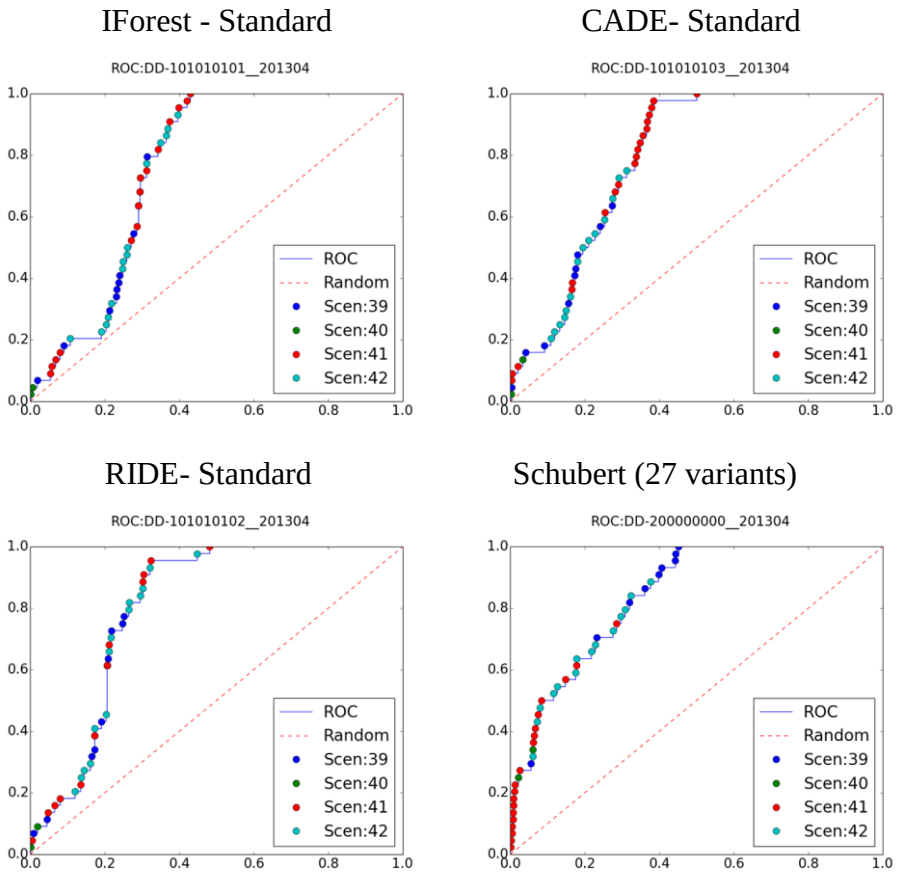


Figure 23. *Preliminary Results of Diversity Experiments*

These experiments were placed on hold pending enhancements to the PRODIGAL framework to allow computation of partial-day and multi-day features. We consider this a promising area for future research.

1.10. Temporal Aggregation Experiments (Leidos SOW 2.2.2)

1.10.1. Background

PRODIGAL scores user days, that is, days in which users are active on their computers. Scoring user days provides the analyst the most unusual individual days for the month, ranked from the highest to lowest (i.e., from the most to the least unusual days). However, we do not systematically apply the user day scores to find which users who repeatedly display the most unusual behavior. For example, ranking user days does not users who had multiple, high-scoring days in a time period whereas an analyst, visually reviewing the output from PRODIGAL's ensemble would likely recognize patterns (e.g., a user who exhibits anomalous behavior on consecutive days, or a week apart on the same day). Our goals in temporal aggregation experimentation were to develop a detector, D, that (1) used output from the ensemble (user ID, rank, and day) to find the most unusual users in a time period and (2) could serve as another detector in the PRODIGAL system.

Our experiments differ from traditional and recent research focused on the assessment of temporal aggregation techniques in the context of time series analysis. Traditionally, this

research has focused on topics in economics and finance such as modeling interest and exchange rates. Recent research has extended the previous work in temporal aggregation time series analysis to agronomy and meteorology and some in the social sciences (e.g., traffic patterns). We believe that in the context of insider threat detection, our research for temporal aggregation is novel.

1.10.2. Methodology

Designing the temporal aggregation model

As the PRODIGAL system consists of over a hundred of features and detectors, we wanted a simple approach for temporal aggregation detector that used information that was already available. Therefore, we used the following output from our ensemble detector: user ID; user day rank (i.e., rank); and date (i.e., day). We combined these outputs into two parameters for temporal aggregation: rank cutoff (τ_1) and the number of days (τ_2) that a user has at a given rank cutoff point. **Table 5** below describes how we set these parameters.

***Table 5**. How we set parameters for temporal aggregation.*

Name	Definition	Possible values
τ_1	The rank of a user day score; interpreted as a cutoff point	The number of user day ranks in the top 5, 10, 20, 50, 100, 200, 500, 1000, 5000, and 10000; 10 values in total
τ_2	The count of the number of user days a specific user has at rank *r*	The number of days in the time period that a user has at or higher than a given rank cutoff point; for a month, 1 – 31

We selected the values of τ_1 primarily for two reasons: first, we believe that analysts expect to see "top *n*" in operational environments and, second, after analyzing the distribution of user days scores, these ranks resulted in near-linear increases in the number of users. As detector parameter τ_2 relates the number of times that a user has a day at a specific rank, the value of τ_2 is the range of days in the time period; thus, in our experiments, the values of τ_2 varied from 1 to 31, depending on the number of days in the month. For a month, there are between 280 and 310 possible detectors.

Specifying the detector

Figure 24 depicts our temporal aggregation model development methodology. For each month, we (1) obtain the count of all distinct user IDs (including RT users), ranks, and days from the ensemble detector. Using those inputs, we (2) find the count of the all users at each value of τ_1 and the count of days for all users at by rank cutoff point (τ_2). We form combinations of each parameter (4) and develop the detectors (D) for the period of analysis; examples of D include: Top 5, 1 Day; Top 10, 3 Days; Top 50, and 4 Days.

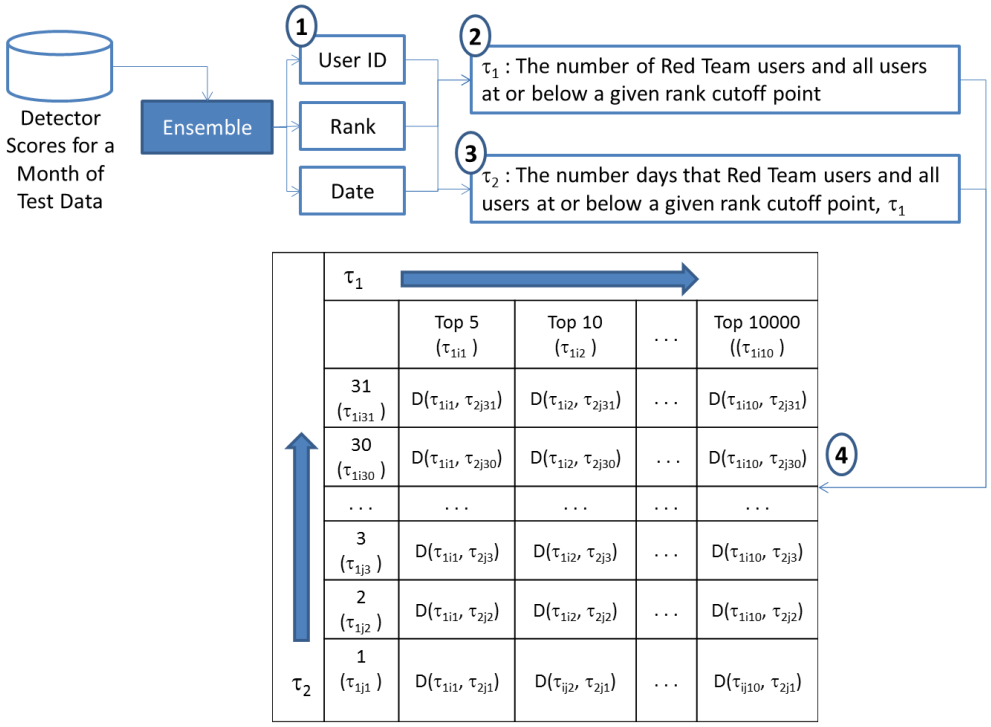


Figure 24. *Our temporal aggregation model development methodology.*

Program Data

We used 21 full months of ADAMS test data (approximately 165,000 total user days scores/month) from September 2012 and July 2014 to populate the model. We aggregated all user behavior for the month and did not distinguish between a Red Team user's "normal" (i.e., the days for which there were no synthetic observables for the Red Team user) and "malicious days" (i.e., days with synthetic observables inserted to simulate insider threat behavior). In an operational context, we believed that this approach was suitable for temporal aggregation as an analyst could, by starting with a higher-ranked day, discover low-signal malicious insider behavior. Output from PRODIGAL's ensemble algorithm served as ground truth for our experiments. Specifically, we used the ensemble ranks by parameter value (e.g., "count of distinct users who have user days at rank equal to or less than 50") and count of days at rank of Red Team user days (e.g., "count as inputs to model parameters, grouping all days within the period by user ID. Ensemble ranks were the input for the model parameter (τ_1) and the count of user days at each rank (τ_2). The count of all RT users per month from the evaluation tea's answer keys provided input to the model's metric, lift (see metrics and evaluation section).

1.10.3. Metrics and evaluation

We used lift, a data mining metric, as the value of the detector, *k*. Lift characterizes the improvement offered by a classifier over random choice, and is an appropriate method to apply in our temporal aggregation. As lift measures the amount of data enrichment offered by a classifier, it enables us to assess the improvement in detecting malicious insiders by looking at focused subsets (e.g., the number of users who have a rank at or above 50 three days in the month) of the overall population. In our experiments, we defined lift as:

$$\frac{\dfrac{\text{The number of Red Team users at } D(\tau_{1i}, \tau_{2j})}{\text{The number of all users at } D(\tau_{1i}, \tau_{2j})}}{\dfrac{\text{The number of all Red Team users in the data set}}{\text{The number of all users in the data set}}}$$

We evaluated the temporal aggregation detector's performance in two ways: (1) average lift across all months and (2) average lift by specific Red Team scenario. In the first approach we calculated the lift by data month (across multiple and different scenarios) and averaged lift of each classifier across all months in the set (i.e., 21 months between September 2012 through July 2014). In the second approach, we calculated lift by each scenario type, averaging lift of each classifier across scenario instance. There are 36 scenarios and 74 distinct data sets in the data range. For example, we averaged lift by classifier for the five instances of the Snowed In scenario, spanning multiple data months (July and October 2013 and July 2014).

Experiment results

Table 6 shows the final results of the model across all months.

Table 6. *Final results of the temporal aggregation model across all months.*

Days	Top5	Top10	Top20	Top50	Top100	Top200	Top500	Top1000	Top5000	Top10000
31										0.000
30								0.000	0.000	0.000
29						0.000	0.000	0.000	0.000	0.000
28					0.000	0.000	0.000	0.000	0.000	0.000
27					0.000	0.000	0.000	0.000	0.000	0.000
26					0.000	0.000	0.000	0.000	0.000	0.000
25					0.000	0.000	0.000	0.000	0.000	0.000
24					0.000	0.000	0.000	0.000	0.000	0.000
23					0.000	0.000	0.000	0.000	2.028	3.031
22					0.000	0.000	0.000	4.391	1.716	1.613
21				0.000	0.000	0.000	0.000	3.991	1.616	1.967
20				0.000	0.000	0.000	0.000	4.803	1.313	1.451
19				0.000	0.000	0.000	0.000	2.689	1.001	1.847
18				0.000	0.000	0.000	0.000	1.880	0.876	3.580
17				0.000	0.000	0.000	0.000	1.566	0.725	2.866
16				0.000	0.000	0.000	0.000	1.314	1.519	2.537
15				0.000	0.000	0.000	0.000	1.164	1.419	2.459
14				0.000	0.000	0.000	0.000	0.885	3.023	2.982
13				0.000	0.000	0.000	7.391	3.999	2.617	2.446
12				0.000	0.000	0.000	5.207	5.196	3.434	2.230
11				0.000	20.828	11.456	6.832	4.750	3.099	2.054
10				**31.242**	17.624	8.486	6.832	3.575	2.872	1.858
9				**20.828**	15.801	7.050	6.547	3.138	2.533	2.063
8			0.000	**20.828**	11.456	7.050	8.072	2.513	2.828	1.994
7			**42.958**	18.329	8.331	6.641	5.726	1.890	2.970	2.008
6			**42.958**	13.886	6.943	12.911	4.028	1.458	2.707	1.954
5		0.000	**34.367**	8.812	6.641	11.900	3.048	1.938	2.882	2.350
4		**114.556**	**20.828**	7.637	6.641	7.781	2.025	3.079	2.753	2.288
3	**91.644**	**29.884**	14.319	6.641	4.981	3.597	1.567	2.970	2.704	2.116
2	**37.153**	12.274	10.108	5.054	4.122	1.862	2.550	4.301	2.341	1.769
1	6.839	4.103	4.238	2.022	3.436	1.918	3.233	3.101	1.883	1.562

Table 7 shows the top ten, most frequently –occurring temporal aggregation classifiers across all months.

Table 7*. The top ten, most frequently –occurring temporal aggregation classifiers across all months.*

Rank	Lift	Detector	Rank	Lift	Detector
1	114.50	Top 10, 4 Days	6	34.367	Top 20, 5 Days
2	91.64	Top 5, 3 Days	7	31.242	Top 50, 10 Days
3	42.958	Top 20, 7 Days	8	29.844	Top 10, 3 Days
4	42.958	Top 20, 6 Days	9	20.828	Top 50, 9 Days
5	31.153	Top 5, 2 Days	10	20.828	Top 50, 8 Days

A review of the most frequently occurring temporal aggregation classifiers suggests that analysts focus on users who are often highly unusual within a given time period. **Table 8** shows the performance of the temporal aggregation detector by scenario, with lift averaged across the distinct instances within a scenario.

Table 8. *Performance of the temporal aggregation detector by scenario, with lift averaged across the distinct instances within a scenario.*

Scenario Name	Count of scenario instances	Averge lift of D across scenario instances
Snowed In	5	1374.667
Anomalous Encryption	2	147.842
Exfiltration Prior to Termination	2	146.842
Selling Login Credentials	1	109.442
Czech Mate	1	51.47
Manning Up Redux	1	36.036
Byte Me	2	30.208
Breaking the Stovepipe	3	25.663
Credit Czech	1	23.89
Blinded Me With Science	1	23.387
Survivor's Burden	3	21.517
Job Hunter	1	21.44
What's the Big Deal	1	16.029
The Big Goodbye	1	12.468
Insider Startup	7	11.488
Bona Fides	2	9.932
Conspiracy Theory	2	9.571
Bollywood Breakdown	1	7.918
Layoff Logic Bomb	2	7.717
Parting Shot	1	6.91
Masquerading 2	2	6.487
Circumventing Sureview	2	6.052
Strategic Tee Time	1	4.747
Indecent RFP 2	2	4.078
Indecent RFP	1	4.078
Passed Over	4	3.761
Exfiltration of Sensitive Data Using Screenshots	3	2.877
Gift Card Bonanza	1	2.82
Byte Me Middleman	2	2.63
Naughty by Proxy	4	2.445
Outsourcer's Apprentice	3	2.445
Hiding Undue Affluence	2	2.22
Parting Shot 2 - Deadly Aim	1	2.22
From Belarus With Love	2	2.205
Manning Up	2	1.99
Panic Attack	2	0.855

Table 9 lists the temporal aggregation detectors that produced the highest and lowest lift values by scenario and relates the number of red team and all users for each detector and the number of all users in the month of the best detector.

Table 9. *The temporal aggregation detectors that produced the highest and lowest lift values by scenario.*

Scenario name	Average lift, best detector	Best detector (D) for this scenario	#/RT users at D	#/All users at D	#/All users for the month
Snowed In	1347.667	Top 5, 3 days	1	1	4124
Anomalous Encryption	147.842	Top 1000, 7 days	1	19	5618

Exfiltration Prior to Termination	146.842	Top 20, 1 day	1	13	4372
Selling Login Credentials	109.442	Top 10000, 23 days	1	4	5691
Czech Mate	51.47	Top 500, 2 days	1	83	4272
Hiding Undue Affluence	2.22	Top 5000, 2 days	1	867	5721
Parting Shot 2 - Deadly Aim	2.22	Top 5000, 2 days	1	635	4230
From Belarus With Love	2.205	Top 10000, 5 days	1	723	4392
Manning Up	1.99	Top 10000, 4 days	1	959	5729
Panic Attack	0.855	Top 10000, 5 days	1	700	4286

1.10.4. **Follow-on work**

We have implemented a temporal aggregation filter in our analyst interface (AI) and intend to present highly-anomalous users identified by the temporal aggregation to CI analysts from the data provider and determine the number of those users whose actions are of interest. Also, in reviewing our results in the data laboratory, we noticed that a high percentage of frequently anomalous users (e.g., users who have multiple days in the top 20 user days) appear to perform tasks associated with job roles and functions categorized as high-risk for insider threat (e.g., system administrators). We will review the highly anomalous users from our temporal aggregation method and determine the percentage of users in the top 20 at 1 and more than 1 day and assess the ability of the temporal aggregation method to identify users who are system administrators.

Algorithm / Evidence Combination and Explanation (Leidos SOW 2.2.2)

1.11. Exploiting a Diverse Ensemble of Detectors

The PRODIGAL system is the result of substantial original research to find relevant anomalies in data, resulting in a number of new methods. Because PRODIGAL was evaluated in a rigorous, consolidated test environment, it became critical to understand how to combine those methods into a single system with good performance and to be able to produce explanations for those results. This section summarizes our research results in these areas on the ADAMS test data.

1.11.1. A Language for Combining Detectors and Specifying User Contexts

Using anomaly detection in PRODIGAL, it is critical to construct *contexts* around individuals. By contexts we mean the linking of users' traits, roles and activities to relevant *baseline populations* and *baseline time periods* to compare against. Useful contexts simultaneously reveal threats and minimize false positives. To do this we developed representations of activities of users and have techniques for designing relevant baselines for specific activity scenarios and we specified them using *Anomaly Detection Language* (ADL). We developed this new language to take advantage of having multiple base detectors in PRODIGAL that can be used together. We also developed it to have a formal way to specify multiple *detectors* using the same base algorithm. We presented this work in [Memory 2013] and we summarize it here.

In anomaly detection, we detect anomalies associated with some *entity*, which may be an individual insider, i.e. system *user*, or the anomalies may be associated with a group of entities, so we adopt the more general *entity extent*. Similarly, the anomaly may be associated with a particular period of time, known as a *temporal extent*, and the combination of these two is an *extent*. The inputs to our analysis are *records* of (trans)actions by entities comprising *values* of fields known as *features*, and the outputs are *scores* on the extents. Additionally, because the analysis is done in stages, scores themselves are treated as features. We use this vocabulary to define the ADL syntax, which is shown in **Figure 25**.

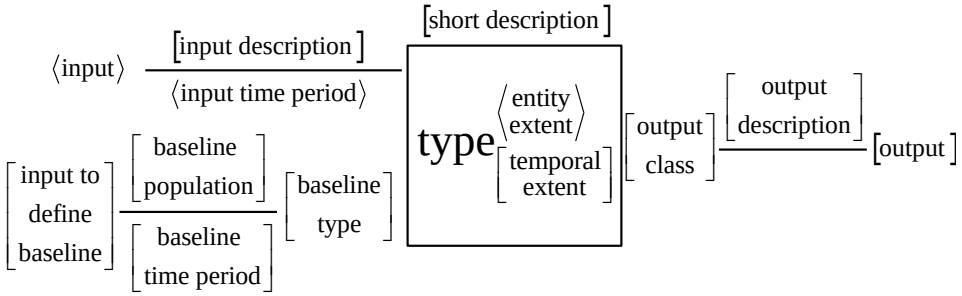


Figure 25. *Anomaly Detection Language (ADL) Syntax*

Components in the language, are rectangles connected by lines representing sets of records passed between them and have types, such as statistical anomaly detector type (denoted by the symbol S), group detector (G) which discovers communities of entities which can be used as baseline populations, classifier (C) which filters and partitions input records, aggregator (A) which summarizes records into features, normalizer (N) which rescales records and features with respect to some context, *AND* and *OR* used when sets of records are joined and contain different values for the same feature, and *union* and *intersection* when no combinations are necessary.

The inputs to the component enter on its left, with the input time period placed below the input line, and the optional baseline drawn as a second line on the left with the baseline population and time period placed above and below that line, respectively. The output entity and temporal extents are super- and sub-scripts on the component type, respectively and exit on the component's right that, when joined, represents a join of the records, in the sense that tables of a database are joined.

If a baseline is provided, a baseline type specifies how the baseline is used by the component. In a cross-sectional baseline (C) entity extents are compared to others within the same temporal extent; in a longitudinal baseline (L) each entity will be handled individually and different temporal extents for that entity are compared to one another; and a simultaneous baseline (S) combines the first two and compares each input extent to all baseline extents. **Table 10.** Expected behavior of an anomaly detector with a variety of inputs and baselines lists the expected behavior of an anomaly detector with a variety of inputs and baselines.

Table 10. *Expected behavior of an anomaly detector with a variety of inputs and baselines*

	Baseline			Anomaly Detector	
Input Time Per.	Population	Time Period	Type	Extent	Scores Each. . .
November	All users	November	Cross-sectional	User-Month	User in November
November	All users	November	Longitudinal	User-Day	Day in November for each user
November	All users	November	Cross-sectional	User-Day	User for each day of November
November	All users	November	Simultaneous	User-Day	User-day in November
November	User roles	November	Cross-sectional	User-Month	User in November for each user role
December	All users	Jan. - Nov.	Simultaneous	User-Month	User in Dec. vs. previous user-months

For example, if the input user data we are analyzing are from the month of November and the baseline population against which we are comparing is all other users from the same month, then the anomaly detector will score each user in November, i.e. each *user-month*, which makes this a cross-sectional baseline.

Whenever a component may output more than one *output class* of records, e.g., a binary classifier has (+) and (-) output classes, they should be placed to the right of the component inside circles connected to output lines, unless only one class of output is needed and that class is clear from context, in which case the output class can be omitted. *Weights* are scalars in the unit interval used to transform features -- usually scores -- and are drawn as the letter *w* inside a rectangle. The type of weighting should be put in a description above the rectangle. Finally, the output of the system is drawn as the letter *O* inside a circle.

Consider a small example in the language, in which we specify a relevant context as features and baselines. In **Figure 26**, we find anomalous users based on the number of blacklist web sites they visit.

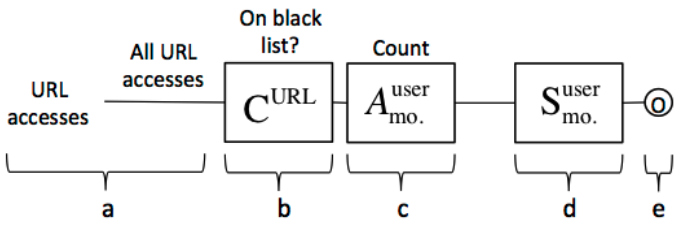


Figure 26*. A small example of ADL.*

We (a) retrieve all the URL access records, (b) keep only accesses to URLs on a blacklist with C^{URL}, (c) count the number of such accesses for each user for each month and (d) run a statistical

anomaly detector over all users in the month using the counts of blacklist URL. Finally, we (e) return anomaly scores for each user. In **Figure 40** we extend the previous example with baseline populations based on the divisions of the organization (b) from user records, e.g., Lightweight Directory Access Protocol (LDAP) (a) in which users are compared cross-sectionally (c).

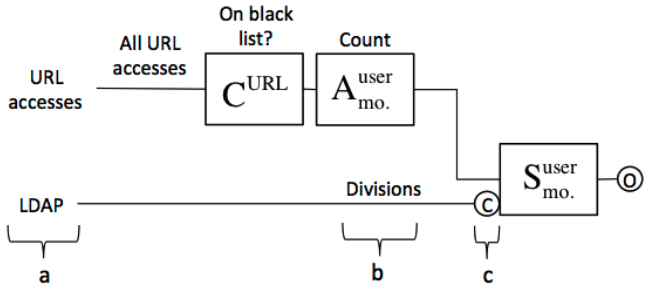


Figure 40. *A small example of ADL, extended.*

Consider a more realistic, but still notional, example. In the *Saboteur* scenario, [Cappelli et al., 2012], shown in **Figure 27**, we look for users with administrator access that could be sabotaging (or preparing to sabotage) systems.

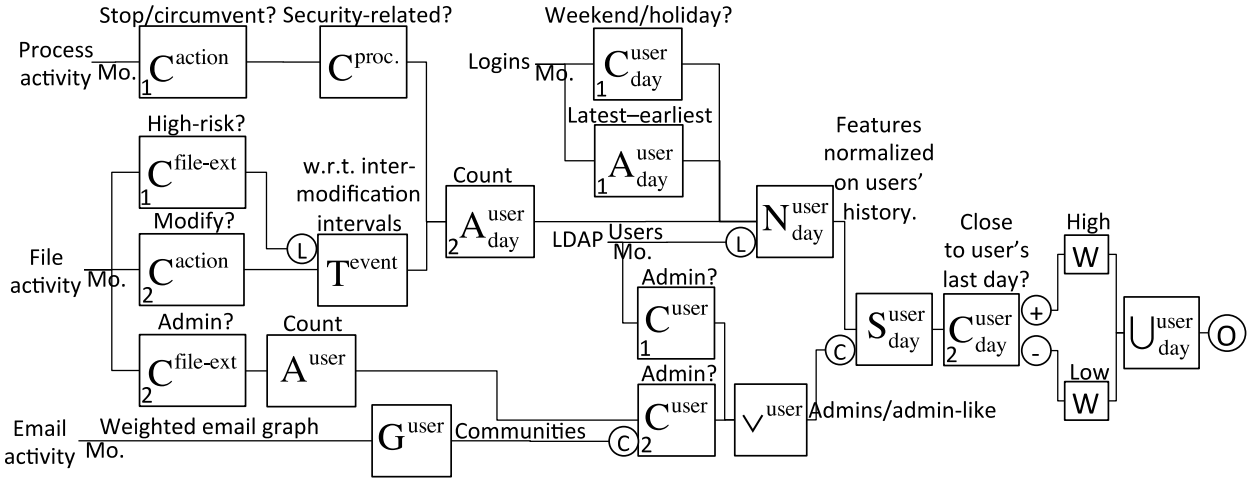


Figure 27. *Example of Saboteur scenario in ADL.*

We count times when security-related processes, e.g., antivirus, are stopped and high-risk files are modified at atypical intervals, add the length of time worked and whether it was a non-work day, then normalize w.r.t. the users' history. To limit the baseline to administrators, we find them in LDAP and combine that set with those who act like administrators according to the file types they access compared to the users' work group in email. Finally, we score the user-days and weight days leading up to a user's departure from the organization more heavily.

The ADL language was used across the PRODIGAL team to specify combinations of detectors and to specify contexts of users that were relevant for finding anomalies associated with insider threats, as seen in other sections of this report. Future research directions would be to implement a parser of the language as an extension of relational algebra, which could then be automatically compiled into PRODIGAL flows.

1.11.2. Unsupervised Ensemble-Based Anomaly Detection

Methods

Having multiple anomaly detection results for the same data naturally leads to the need to combine those results to take advantage of the distinct perspectives embodied in different detectors. Further, an analyst responsible for insider threat detection desires a single ranked list rather than many different result sets from different detectors whose detailed operation he/she may not fully understand. Anomaly detector ensembles combine the results (i.e., scores) from multiple detectors in a way that is analogous to how classifier ensembles combine predictions from multiple classifiers [Dietterich 2000].

Because when building ensembles, we assume that we do not have access to ground truth, it is not known whether one of the individual detectors always performs well; in fact, in our experiments where we have ground truth we found that the best detector varied across data sets. Therefore, one goal of ensemble building is to perform at least as well as the best detector. It is also possible for an ensemble to outperform all of the individual detectors, which is analogous to how some classifier ensembles are able to outperform their individual classifiers.

Selecting an approach for building ensembles depends upon the types of detectors that are used. If all the detectors share an underlying model, then the ensemble approach can leverage that commonality to improve performance, e.g., the method reported in [Lazarevic 2005] varies the features used as input to a single anomaly detection model to build an ensemble. Another way of leveraging a common model is to use the same input features, but alter hyperparameters, which determine how the model is built in each detector [Lazarevic 2005], [Dietterich 2000].

If, however, the detectors do not share a common underlying model then the ensemble-building approach may only assume that the scores from the detectors are given as input, i.e., the features used as input to detectors and the hyperparameters of the detectors are unknown. Because our individual detectors employ a variety of models, we chose an approach that is consistent with this setting [Schubert 2012]. The following paragraphs describe the approach.

Some approaches for ensemble building, including the method we used, employ the following two high-level heuristics. First, if a consensus about which points are most anomalous can be drawn from the individual detectors, then that consensus should be preserved in the final ensemble. Second, because each individual detector is subject to unavoidable biases stemming from the choice of model, choice of input features, hyperparameter settings, etc., the ensemble should prefer combinations of results from detectors with uncorrelated biases.

These heuristics are implemented in two distinct phases in this method. In the first phase they extract a consensus across all detectors from the union of the top k most-anomalous points from each detector. All points in this union are given a score value of 1 and all others are given a score of 0. We chose a value for k for each dataset that included the top 1% of the points. The method then initializes the ensemble with scores from the detector that is most correlated with the consensus. The correlation between detectors and the consensus is found by viewing each as n-length vectors of scores, where there are n points in the dataset, and then using a simple correlation metric to compare the vectors. We used the Pearsons r correlation metric for this.

In the second phase, the method greedily selects candidate detectors to combine with the initial ensemble by preferring the detectors that are least correlated with the current ensemble. The same correlation metric used before is used again here. The candidate detectors scores are

combined with the current ensemble using a point-wise combination function. For this the method uses the average over scores for each point; we also experimented with other functions including the maximum of scores. The algorithm proceeds to accept a candidate detector if the resulting ensemble is no less correlated with the consensus than the previous ensemble; if it is, then the candidate detector is discarded. This phase continues until all detectors are either accepted or discarded.

Results and Discussion

Our initial experiments evaluated our detection results based on our ability to detect user-days with red-team inserted activity. We used a wide variety of detectors, described in reference [Senator 2013], and the ensemble technique described above. Results are summarized in **Table 11**. For each month, we report the area under the ROC curve for our best detectors, for the ensemble, and the ratio of the two, indicating how close the ensemble came to the best. The AUCs reflect the ability of the detectors to find all of the user-days of the union of all scenarios present in the month. These results illustrate that the unsupervised ensemble-based anomaly-detection technique had performance that is close to that of the best of the individual anomaly detectors. The AUC for the ensemble technique was within 5% of the AUC of the best detector, with a 95% confidence interval of +/- 1.5% around a linear regression fit of ensemble to best AUC over 54 individual RT scenario instances. Interestingly, the ensemble-based technique appeared to have results that were similar across datasets, while the best detector varied widely. **Figure 28** illustrates the differences in performance between the ensemble and the best-performing detector for selected months. These 8 months have been selected to show instances when a wide variety of detectors were best performer.

Table 11. *Ensemble and Best Detector Results by Month.*

Month	*Ensemble AUC*	*Best Detector*	*Best Det. AUC*	*Ens. / Best RT*	*Scenarios*
12-Sep	0.8973	CADE:UP	0.9703	92.47%	Circumventing SureView Insider Startup
12-Oct	0.9319	CADE:UP	0.9804	95.05%	Insider Startup
12-Nov	0.7542	File	0.7895	95.53%	Anomalous Encryption, Layoff Logic Bomb, Masquerading 2
12-Dec	0.8646	GMM:QD	0.8677	99.64%	Anomalous Encryption, Layoff Logic Bomb, Outsourcer's Apprentice
13-Jan	0.8594	RIDE:RD	0.9015	95.34%	Hiding Undue Affluence, Outsourcer's Apprentice, Survivor's Burden
13-Feb	0.7632	EGMM:QD	0.7793	97.94%	Bona Fides, Manning Up, Survivor's Burden
13-Mar	0.8853	IFOR:QD	0.8963	98.77%	Bona Fides, Hiding Undue Affluence, Manning Up Redux
13-Apr	0.8635	RIDE:QD	0.8619	100.19%	Circ. SureView, Indecent RFP, Selling Login Cred., Survivor's Burden
13-May	0.8469	PDE:UP	0.9718	87.14%	Credit Czech, Exfiltration Prior to Termination
13-Jun	0.8852	IFOR:QD	0.9103	97.24%	Czech Mate, Exfiltration of Sensitive Data Using Screenshots
13-Jul	0.8498	RIDE:RD	0.8769	96.90%	Breaking the Stovepipe, Snowed In
13-Oct	0.8938	GMM:RD	0.8972	99.62%	Breaking the Stovepipe, Snowed In
13-Nov	0.8479	RIDE:RD	0.8459	100.23%	Byte Me, Naughty by Proxy
13-Dec	0.8034	EGMM:RD	0.828	97.02%	Byte Me Middleman, Indecent RFP 2, Passed Over
14-Jan	0.8425	IFOR:RD	0.8242	102.22%	From Belarus With Love, Passed Over, What's the Big Deal
14-Feb	0.847	GFADD:84-88-0	0.9775	86.65%	Bollywood Breakdown, Breaking the Stovepipe, Gift Card Bonanza, Naughty by Proxy

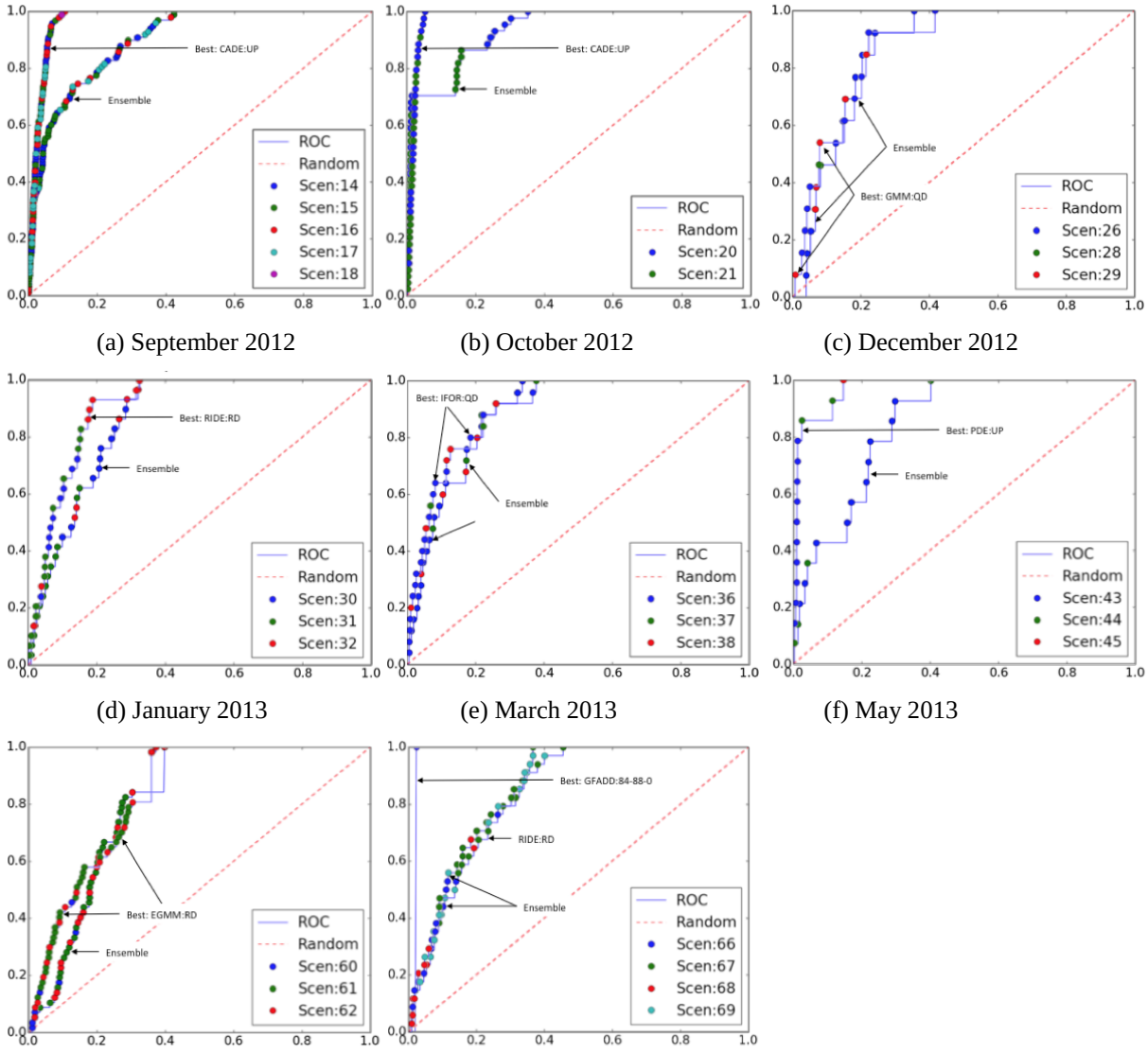
In September and October 2012 (**Figure 42**(a) and **Figure 42**(b)) there were 6 instances of the same RT scenario (14-17, 20-21), and one of the CADE detectors was best both times. Although neither CADE nor the Ensemble were able to score all inserted user-days very high, all inserted scenarios had several user-days in very high rankings, supporting the expectation that a later stage of analysis would easily find the rest given these leads.

Figure 28(c) through e and g show months where different detector models performed best. In each case the Ensemble was able to approximate performance of the best detector. These months contained 12 instances of 10 different RT scenarios.

Figure 28(f) shows Ensemble performance significantly lower than the best detector, although the top few user-days are ranked highly by both. However, this detector, PDE:UP, averaged only 85% of the Ensemble's AUC values over all 16 months, and so could not be relied upon for consistent detection. This is the case for all detectors of all three types and is the principal reason why even an ensemble method that fails to improve on the best score is still valuable.

Finally, we note for the month shown in **Figure 28**(h), the best detector – GFADD with no grid over the feature pair, # file events on removable drives vs. # distinct removable drives – only returns a score for user-days it finds anomalous, so the value of an AUC is questionable. The

second best detector is also shown. Again, the Ensemble comes close, in fact exceeding its AUC value slightly.



(a) September 2012 (b) October 2012 (c) December 2012

(d) January 2013 (e) March 2013 (f) May 2013

(g) December 2013 (h) February 2014

***Figure 28**. ROC curves vs. all RT inserts for the Ensemble and the best detector for various months.*

***Table 12.** Comparison of Scenario-Based Detector to Ensemble Performance, by Red Team Scenario Inserts*

Month	Inserted RT Scenario	Ensemble AUC on RT Inserts	Relevent Scenario Detector	Decenario Detector AUC	Ratio (Ensemble / Insert AUC)
12-Sep	Insider Startup	0.8731	Amb. Lead.	0.9176	95%
12-Sep	Insider Startup	0.88	Amb. Lead.	0.6887	128%
12-Sep	Insider Startup	0.9257	Amb. Lead.	0.9117	102%
12-Sep	Insider Startup	0.9097	Amb. Lead.	0.8389	108%

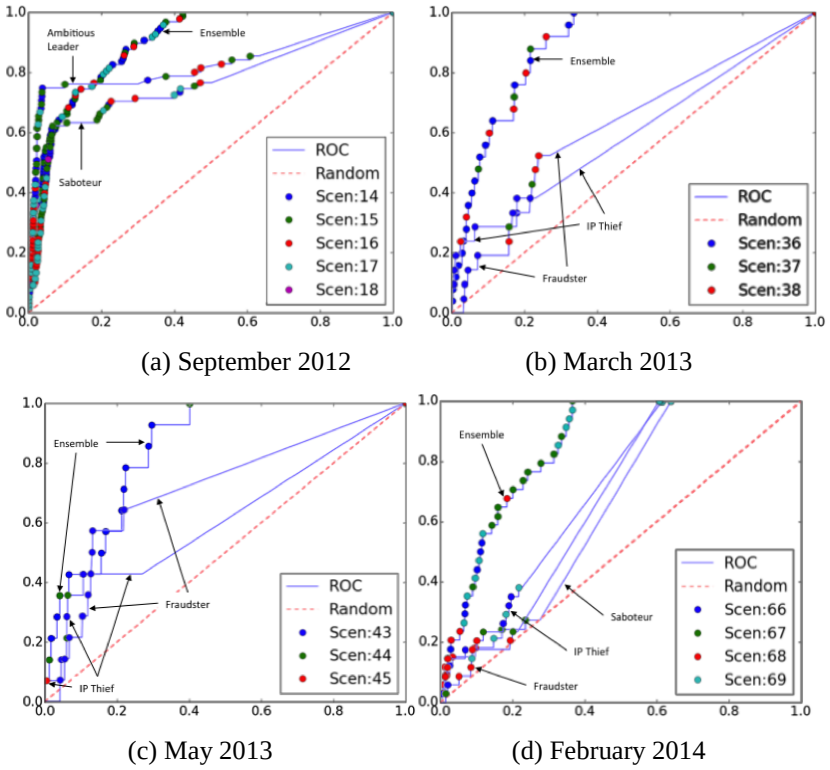
12-Sep	Circumventing SureView	0.9789	Saboteur	0.9465	103%
12-Oct	Insider Startup	0.9171	Amb. Lead.	0.9258	99%
12-Oct	Insider Startup	0.9466	Amb. Lead.	0.8233	115%
13-Mar	Manning Up Redux	0.9041	IP Thief	0.7575	119%
13-Mar	Hiding Undue Affluence	0.8458	Fraudster	0.8115	104%
13-Mar	Bona Fides	0.8447	IP Thief	0.6571	129%
13-Apr	Survivor's Burden	0.8806	Saboteur	0.687	128%
13-Apr	Selling Login Credentials	0.8594	Amb. Lead.	0.6166	139%
13-Apr	Indecent RFP	0.8405	Fraudster	0.6542	128%
13-May	Credit Czech	0.8314	Fraudster	0.8594	97%
13-May	Exfiltration Prior to Termination	0.8476	IP Thief	0.808	105%
13-May	Exfiltration Prior to Termination	0.9994	IP Thief	0.9988	100%
13-Jun	Exfiltration of Sensitive Data Using Screenshots	0.8114	IP Thief	0.4703	173%
13-Jun	Exfiltration of Sensitive Data Using Screenshots	0.9935	IP Thief	0.9769	102%
13-Jun	Exfiltration of Sensitive Data Using Screenshots	0.9251	IP Thief	0.4703	197%
13-Jun	Czech Mate	0.8702	Fraudster	0.6021	145%
13-Jul	Breaking the Stovepipe	0.8176	IP Thief	0.8017	102%
13-Jul	Snowed In	0.8578	Fraudster	0.7243	118%
13-Oct	Snowed In	0.8645	Fraudster	0.8245	105%
13-Oct	Snowed In	0.927	Fraudster	0.694	134%
13-Oct	Snowed In	0.8706	Fraudster	0.8121	107%
13-Oct	Breaking the Stovepipe	0.9313	IP Thief	0.8403	111%
13-Nov	Naughty by Proxy	0.7904	Saboteur	0.5711	138%
13-Nov	Naughty by Proxy	0.718	Saboteur	0.4872	147%
13-Nov	Byte Me	0.8607	Fraudster	0.7205	119%
13-Nov	Byte Me	0.8872	Fraudster	0.7387	120%
13-Dec	Indecent RFP 2	0.9449	Fraudster	0.7217	131%
13-Dec	Byte Me Middleman	0.8279	Fraudster	0.6246	133%
13-Dec	Passed Over	0.7451	Saboteur	0.4951	150%
14-Jan	Passed Over	0.8571	Saboteur	0.6602	130%
14-Jan	What's the Big Deal	0.903	Careless	0.6938	130%
14-Jan	From Belarus With Love	0.7206	IP Thief	0.6269	115%
14-Feb	Bollywood Breakdown	0.9286	IP Thief	0.7904	117%
14-Feb	Gift Card Bonanza	0.7705	Fraudster	0.6438	120%
14-Feb	Breaking the Stovepipe	0.9497	IP Thief	0.835	114%
14-Feb	Naughty by Proxy	0.812	Saboteur	0.5742	141%

Additionally, we see that the ensemble generally outperformed the scenario-focused detectors, including the scenario-focused detectors that we determined later to have been a likely fit to the red team scenario that was actually inserted. In

Table *12* we see AUC performance on individual inserted scenario instances. Ensemble AUC over these instances averages 0.87 and consistently outperforms the selected relevant scenario-focused detector; in 3 of the 41 by over 150%. There is considerable variation in the results, even over instances of the same RT scenario, suggesting the inherent variability of the RT threat simulation process.

Table *12* reports results on all months where the comparison was possible, not only months where the ensemble performed well. **Figure 29** depicts the performance of the ensemble method compared to the corresponding scenario-focused detectors for 4 of these months. **Figure 29**(a) shows a month where the inserted scenarios and the scenario-base detectors correspond fairly well. The typical response of the scenario detector is to score some of the best matching user-days well, often better than the Ensemble, but then drop off rapidly in the remainder of inserted targets. IP Thief in **Figure 29**(a) through **Figure 29**(d) shows some of this behavior, although the presence of other scenarios in the month masks its performance on the relevant inserts. We have reviewed individual scenario-based detectors on separate inserted RT scenarios and see the same response, but the data are too cumbersome to present graphically in this paper. However, it is worthwhile to note the number of instances in

Table *12* where scenario-based detectors performed close to the Ensemble cases where the detector does well on a few high ranking user-days at the expense of not identifying others. We are investigating ways to make use of these targeted responses.



(a) September 2012 (b) March 2013

(c) May 2013 (d) February 2014

Figure 29. *ROC curves for the Ensemble and the Scenario-Based Detectors for various months.*

Figure 44 identifies the sets of detectors selected by the ensemble each month and compares them to the best performing detectors for that month. The best-performing detector was included in the ensemble in only 4 of the 22 months of data. And because in those months there are on average more than six detectors selected for the final scoring step, and all ensembles comprise equally-weighted detectors, the best detector is never given more than one sixth of the weight in this ensemble result. Therefore, the ensemble technique is able to achieve comparable performance to the best detector by combining detectors and with those detectors often excluding the best

performing detector. Recall that the heuristics followed by the ensemble technique favor detectors that are either most close to the consensus or those that are able to add diversity to the ensemble (least correlation with the ensemble) without reducing correlation with the consensus. Thus in these data sets the best-performing detector generally disagrees with the consensus from other detectors, yet a combination of those other detectors can be built automatically that performs nearly as well as that best detector.

Detectors	Sep-12	Oct-12	Nov-12	Dec-12	Jan-13	Feb-13	Mar-13	Apr-13	May-13	Jun-13	Jul-13	Oct-13	Nov-13	Dec-13	Jan-14	Feb-14	Mar-14	Apr-14	May-14	Jun-14	Jul-14	Sep-14
CADE:R							E	E														
CADE:UP	B	B																				
EGMM:RD									E			E	E	B	E	E				E		E
EGMM:QD						B																
GFADD:84-85-0																		B	E			
GFADD:84-88-0																B			E	E	E	
GFADD:83-84-8						E																
GFADD:83-85-0	E	E	E		E	E													E	E		E
GFADD:83-85-8		E		E	E	E				E												
GFADD:84-85-8		E	E	E	E	E						E	E	E			E		E	E		
GFADD:84-88-8		E						E				E										
GFADD:89-90-8		E	E		E	E													E	E		E
GFADD:85-88-0																						
GFADD:92-95-0	E	E	E		E	E						E							E	E		E
GMM:RD		E	E	E		E						B			E	E						
GMM:QD				B																		
IFOR:RD							E	E	E	E	E	E	E	E	EB*	E	E	E	E		E	E
IFOR:QD							B			B										E		
PDE:R10K	E	E	E	E	E	E				E	E	E							E	E	E	E
PDE:UP	E	E	E	E	E	E			B	E	E	E	E		E	E	E		E	E	E	E
PDE:UP10K											E					E		E				
RIDE:QD	E	E	E	E	E	E		EB*		E	E	E	E		E			E	E		E	E
RIDE:RD	E	E			EB	E	E				B		EB*			E	B	E	B	B	B	
VSM							E		E	E	E	E	E	E	E	E	E	E	E	E	E	E
Indicator:File			B																			B

(E = included in ensemble final selection; B = Best individual AUC; B = Next best, after ensemble AUC)*

Figure 30. *Ensemble Composition and Best-Performing Detector, by Month*

In 10 out of 16 months at least one of the accepted detectors used an underlying model that was shared with the best-performing detector. For example, in Dec-12 the best performing algorithm is GMM Density Estimation via unusualness of counts vs. company, which shares the same underlying model Gaussian mixture models as one of the accepted detectors, GMM Density Estimation using Raw Counts; the difference between these two detectors is the method of normalizing input features, which we mentioned as an important element of detector configuration and a source of diversity in our detection suite.

These apparently contradictory results, good performance of the ensemble without participation of the best detector in the final scoring step, led us to devise a series of simulation experiments to thoroughly understand the role of the participating detectors in ensemble methods. It is clear that

the best detector plays a role in driving ensemble performance. When the best detectors perform poorly, so does the ensemble. So their role in constructing the consensus is important. We aim to improve ensemble performance significantly through this research by applying the two heuristics more effectively as well as through continued improvement of the individual detectors.

Finally, **Table 13** displays the results of running more ~40 detectors, and the Ensemble, over 22 months of live data, showing the AUC achieved on each inserted red team scenario.

Table 13. *Performance by Scenario Instance of PRODIGAL's Detectors over 22 months.*

Scenario	TBAD	VSM (D)	GFADD (92-95-0)	GFADD (95-96-8)	GFADD (89-90-0)	GFADD (89-90-8)	GFADD (82-84-8)	GFADD (83-85-0)	GFADD (83-85-8)	GFADD (83-84-8)	GFADD (85-88-0)	GFADD (84-88-0)	GFADD (84-88-8)	GFADD (84-85-0)	GFADD (84-85-8)	GMM (RD)	GMM (QD)	EGMM (RD)	EGMM (QD)	RIDE (RD)	RIDE (QD)	IFOR (RD)	IFOR (QD)	CADE (ALPH-R-10k)	CADE (ALPH-UP-10k)	CADE (UP)	PDE (ALPH-R-10k)	PDE (ALPH-UP-10k)	PDE (ALPH-UP)	Saboteur (2)	IP Thief (1)	Fraudster (1)	Amb. Lead. (1)	File (1)	URL (1)	File-URL (1)	URL, File-Log.	Careless	Rager	ENSEMBLE
Insider Startup	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Insider Startup	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Insider Startup	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Insider Startup	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Circumventing Sureview	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Insider Startup	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##			##
Insider Startup	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##			##
Anomalous Encryption	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##			##	##	##	##	##	##					##	##	##	##			##
Masquerading 2																##	##	##	##	##	##			##	##	##	##	##	##						##	##	##			##
Masquerading 2	##															##	##	##	##	##	##			##	##	##	##	##	##						##	##	##			##
Layoff Logic Bomb	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##			##	##	##	##	##	##					##	##	##	##			##
Layoff Logic Bomb			##	##			##	##	##	##	##			##	##	##	##	##	##	##	##			##	##	##	##	##	##					##	##	##	##			##
Anomalous Encryption			##	##			##	##	##	##						##	##	##	##	##	##			##	##	##	##	##	##					##	##	##	##			##
Outsourcer's Apprentice	##		##	##			##	##	##	##						##	##	##	##	##	##			##	##	##	##	##	##					##	##	##	##			##
Outsourcer's Apprentice			##	##			##	##	##	##						##	##	##	##	##	##			##	##	##	##	##	##											##
Survivor's Burden	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##			##	##	##	##	##	##					##	##	##	##			##
Hiding Undue Affluence	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##			##	##	##	##	##	##						##	##	##			##
Survivor's Burden	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##			##	##	##	##	##	##					##	##	##	##			##
Bona Fides	##															##	##	##	##	##	##			##	##	##	##	##	##						##	##	##			##
Manning Up	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##			##	##	##	##	##	##					##	##	##	##			##
Manning Up Redux	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Hiding Undue Affluence	##		##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Bona Fides	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Survivor's Burden	##		##	##			##	##	##	##	##			##	##	##	##	##	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Circumventing Sureview	##		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##		##	##	##	##	##	##	##	##	##		##	##	##	##	##	##	##	##	##	##
Selling Login Credentials	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Indecent RFP	##		##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##		##	##	##	##	##	##	##	##	##	##
Credit Czech	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
filtration Prior to Termination		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
filtration Prior to Termination		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Exfiltration of Sensitive Data	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Exfiltration of Sensitive Data			##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Exfiltration of Sensitive Data	##		##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Czech Mate			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Breaking the Stovepipe	##	##	##	##			##	##	##		##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Snowed In	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Snowed In	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Snowed In	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Snowed In	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Breaking the Stovepipe		##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Naughty by Proxy	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Naughty by Proxy	##	##	##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Byte Me	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Byte Me	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Indecent RFP 2	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Byte Me Middleman	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Passed Over	##	##	##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Passed Over	##	##	##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
What's the Big Deal	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
From Belarus With Love	##	##	##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Bollywood Breakdown	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Gift Card Bonanza	##	##	##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Breaking the Stovepipe	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Naughty by Proxy	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Outsourcer's Apprentice	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Manning Up	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Panic Attack	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Parting Shot	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Passed Over	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Conspiracy Theory	##	##	##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Job Hunter	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Naughty by Proxy	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Indecent RFP 2	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Parting Shot 2 - Deadly Aim	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Strategic Tee Time	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Conspiracy Theory	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Panic Attack	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
The Big Goodbye	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Blinded Me With Science	##	##	##	##			##	##	##	##						##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Byte Me Middleman	##	##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Snowed In	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
From Belarus With Love		##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Insider Startup		##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##
Passed Over		##	##	##			##	##	##	##	##			##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##	##

1.11.3. A New Ensemble Using Explanations

Like other systems using anomaly detection, the PRODIGAL system includes a number of base anomaly detectors and combines them into an ensemble. So far, much of the prior work using these ensembles has used a limited number of base detectors and, for good reason, the detectors shown to work well across many problem domains are the first to be picked, e.g., local outlier factor and Isolation Forest. For PRODIGAL, we had to expand the set of base detectors beyond those tried and true detectors to be able to handle, for example, detectors that are specialized for a particular insider threat scenario. When we do this, however, it is important to know whether the ensemble model we use can accommodate a larger number of detectors and to know what will happen if some of those detectors have lower accuracy than the typical set of base detectors.

To study this problem, we studied whether the [Schubert 2012] ensemble method is robust to additional base detectors. In the process, we developed a new ensemble model using detector explanations as input that is more robust to errors in base detectors. In our tests, the new model showed potential for the robustness needed to permit a broader range of base detectors than was possible previously. We presented our results in [Memory 2015], which we summarize here.

For our experiments, we used PRODIGAL's base detectors, which have the ability to generate anomaly scores and also explanations. To determine whether or not PRODIGAL's base detectors with varying accuracy reduce the accuracy of the ensemble model using them, we considered three different combinations of synthetic, erroneous base detectors, which we added to an ensemble to test its robustness. For the first combination, we *random* base detectors, which have random and unrelated scores and explanations. The scores are drawn uniformly from [0,1]. The explanations are drawn from [0,1] in D dimensional space, i.e., there are D features drawn from the dataset. A second combination occurred when we added *copies* of the same base detector; in this case, all scores for the same point were equal and all explanations for the same point were identical. Our third combination consisted of adding detectors that varied only by score or by explanation. Our goal was to find an ensemble model that is robust to each of those forms of synthetic base detectors.

We considered four preferences to guide the operation of the new ensemble model.

Agreement. For each point, the model will only select detectors that have similar explanations for that point. Conceptually, a perfectly normal point has no explanation but we force one to be generated and it can be heavily influenced by noise. So, agreement should be unlikely for normal points and more likely for anomalous points.

Independence. The model will not treat base detectors that always agree as independent sources of information. In an extreme case that we consider, base detectors can be identical and their agreement will not be an indication of a true anomaly. Base detectors with overall agreement on explanations are *dependent* detectors.

Plurality. Except in the case of dependent detectors, a grouping of multiple, tightly agreeing detectors is less likely to happen by chance than a single pair in agreement. Having a pair of agreeing detectors is likely if detectors are numerous (cf. the birthday paradox). As a result, a *plurality* of independent and agreeing detectors is an indication of a true anomaly. A pair of detectors is more likely to be selected if, all else being equal, there exist two other detectors having explanations that agree with the first pair.

Strongest Response. Among selected detectors for a point, choose the strongest response -- the highest score -- as the final score for that point.

As we have done for other PRODIGAL experiments, we measured the performance of ensemble models using the benchmark from OSU. We used four randomly selected problems from each available difficulty level of the *abalone*, *concrete*, *fault*, *imgseg*, *opt.digits*, *pageb* and *yeast* datasets, totaling 84 problems.

We chose multiple configurations of two PRODIGAL base anomaly detection algorithms with two goals in mind: (1) some base detectors perform well and others do not. (2) There is varying dependence among detector pairs; some are generally dependent and others are not. The two algorithms we use were Isolation Forest and Repeated Impossible Discrimination Ensemble (RIDE). We chose three configurations of Isolation Forest by setting the size of the random forest used in the detector. We also chose three configurations of RIDE by setting the epsilon parameter.

We used two versions of the new ensemble model. The first version (E1) used all four preferences while the second (E2) removed the independence and plurality preferences. We compared against the [Schubert 2012] ensemble. We also compare against a simple ensemble in which the final score for a point is the average of scores for the point from all base detectors (A).

For each ensemble, we found the area under the receiver operating characteristic (ROC) curve (AUC) for each of the 84 problems with only the real base detectors included, then with 2, 10 or 100 additional synthetic base detectors. **Figure 31** shows the robustness of ensembles to synthetic base detectors with random scores.

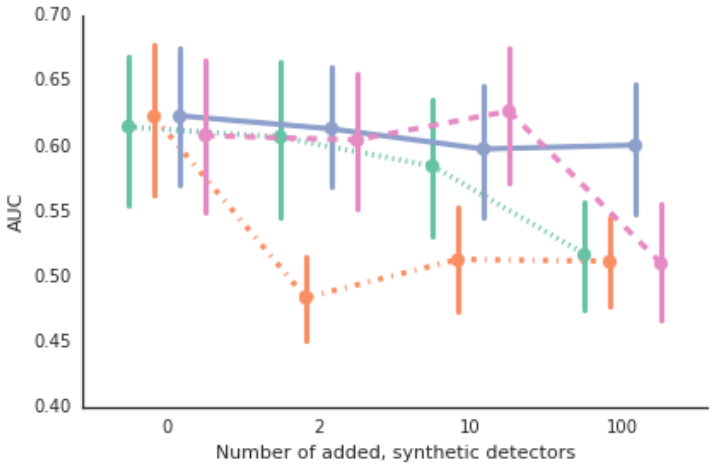


Figure 31. *Robustness of ensembles to synthetic base detectors with random scores*

As the number of additional base detectors increases, the area under the ROC curve (AUC), shown on the vertical axis, for the greedy ensemble G (purple dash) eventually falls. The explanation-based ensemble E1 (blue solid) is relatively unaffected, while the average-based ensemble A (green dotted) and E2 (orange dot-dash) are strongly affected. To examine the effects of scores and explanations separately, this configuration started with a set of identical, erroneous detectors then varies their scores only. The G ensemble is not robust for this case, as its accuracy is ultimately affected by the random scores.

Figure 32 shows robustness to detectors with randomly varying scores and explanations.

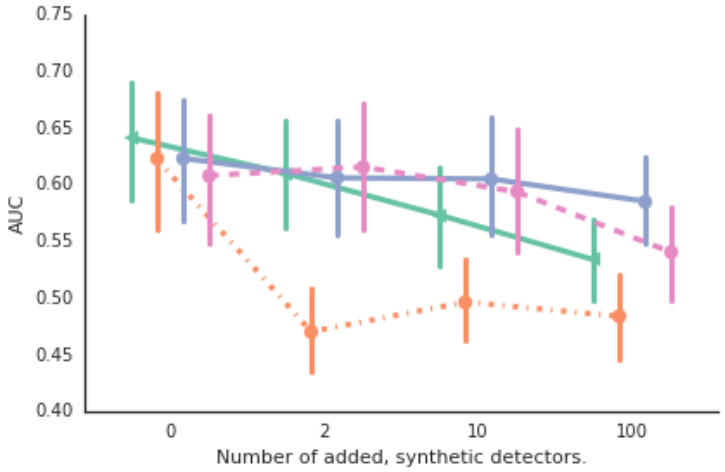


Figure 32. *Robustness to detectors with randomly varying scores and explanations*

As we added base detectors, the AUC for all ensembles decreases, with E1 (blue solid, circle markers) least of all. A variant of E1 without the plurality preference (green solid, triangle markers) reveals the effect of plurality on robustness. G is again affected by the random scores; E1 would also be strongly affected by this case due to the random explanations (given enough random explanations, some are likely to agree); however, the preference for plurality helped E1 be robust for this case. The figure also shows a variant of E1 (solid triangle markers) that has the plurality preference disabled; the variant is less accurate than E1, as expected.

With this new ensemble, we showed important ways that explanations could improve robustness of the ensemble over the ensemble we have used in PRODIGAL. A future direction for research would be to study whether this ensemble approach could produce a combined list of explanations in addition to the combined list of scores.

1.12. Explanations

During Phase 2, the PRODIGAL team directed significant effort towards developing methods of explaining the results of anomaly detection, presenting those explanations to human analysts, and using the explanations to improve downstream automated processes, such as ensemble score combination.

1.12.1. A Common, Understandable Representation for Anomaly Explanations

In PRODIGAL, explanations of an anomaly score are defined as lists of the domain-knowledge-based features with associated weights. This was chosen as the simplest representation of knowledge about the scored entity-extents common to all detection algorithms. Furthermore, by explaining anomaly scores in terms of features derived from known activities, an analyst does not have to understand details of the algorithms.

We conducted a study of the utility of feature-based explanations, which is summarized in Section 1.12.2 and included in [Goldberg 2016].

Level 0 Explanations

We implemented explanations in a staged approach, beginning with a simple enumeration of the outlier scores of pre-computed features. Each entity receiving a score in PRODIGAL has associated with it a large number of feature values, V(U,D,F), where U = user ID, D = date, and

F = feature ID. PRODIGAL computes a statistical outlier score for each value by comparing it against all other users for that date, the comparison population being {V(x,D,F)}. We compute an outlier score using the cdf of the logistic distribution with mean and variance of this population. This score is normalized to [0,1] and is easily compared with other features' scores. We might think of this score as the pseudo-likelihood of outlier status since it estimates the likelihood that the feature value is greater than others from the base population. A Level 0 explanation of a scored entity is a weighted list of features, where weights are equal to the outlier scores.

Pre-computed features were selected from a wide range of user behaviors identified by intelligence analysts, and thus represent specific behaviors meaningful to an analyst. Some examples include: URL upload count, email event count, average recipient count per email sent, fixed drive file event count, and upload /distinct URL domain ratio. A sample of the values and outlier scores computed for the last example is shown in **Figure 33** and **Figure 34**.

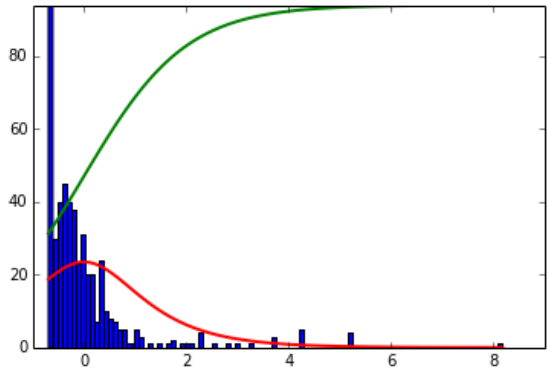


Figure 33. *Histogram of values for the feature: Upload / Distinct URL domain ratio*

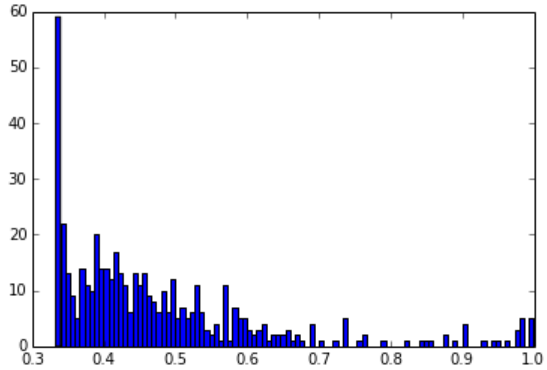


Figure 34. *Histogram of outlier scores for the feature: Upload / Distinct URL domain ratio*

For these examples of pre-computed features, the first histogram shows the raw feature value with the density and cdf of the fitted logistic distribution. The second histogram shows the resulting outlier scores.

Level 1 Explanations

A number of anomaly detection algorithms incorporated into PRODIGAL were modified to produce explanations specific to their own score, their estimation of the anomalousness of each entity-extent. Four general approaches were explored. (Note: $f(x_i)$ below is the learn "normal"

density of entity-extents.)

Sequential Marginal (SeqMarg)

- Choose First feature i that minimizes $f(x_i)$
- Choose Second feature j that minimizes $f(x_i, x_j)$
- etc.

Independent Marginal (IndMarg)

- Order features according to increasing $f(x_i)$
- I.e. order according to independent anomalousness of each feature

Independent Dropout (IndDO)

- Order features according to decreasing $f(x_{-i})$
- I.e. order according to how much more normal x looks after removal

Sequential Dropout (SeqDO)

- Select first feature i as one that maximizes $f(x_{-i})$
- Select second feature j as one that maximizes $f(x_{-i-j})$
- etc.

Figure 35 shows results of a comparative study in which these four methods were used to produce a "sequential feature explanation (SFE)". A SFE is an explanation in which the features are ordered by importance to the anomaly detector. A simulated expert was presented with one feature at a time, until it was able to detect anomalies.

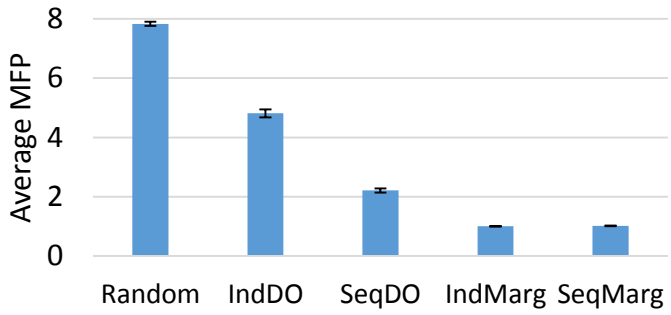


Figure 35. *Results of a comparative study in which four explanation methods were used to produce a "sequential feature explanation (SFE)".*

Combining Explanations

The most effective available method for generating Level 1 explanations was implemented in PRODIGAL anomaly detectors where computationally feasible. The PRODIGAL team developed mechanisms for combining and presenting these Level 1 explanations from the detectors constituting the system's detection ensemble. The Analyst's Interface can display, not only the single feature outlier scores described earlier, but also these combined explanations from the detectors chosen to represent PRODIGAL's best ensemble.

In addition to combining Level 1 features as output from the ensemble process, research was undertaken to develop methods that use component algorithms' explanations as input to the ensemble process.

1.12.2. Evaluating Explanations (Leidos SOW 2.2.1)

As described in Section 1.12.1, explanations of anomaly scores in PRODIGAL are drawn from single feature outlier scores or generated by anomaly detection algorithms. In either case, they

comprise a list of weights associated with a large number of, frequently inter-dependent, low level features. While these features are derived from domain knowledge about potential insider threat activity, they may be confusing as well as redundant - giving the analyst too much detail to get a good picture of where to look.

We explored the idea of transforming PRODIGAL's explanations to a vector of weights on analyst-originated explanations (which we'll call labels to distinguish them from features). We learned a transformation from the feature vectors to label vectors over a set of manually labeled user-days. Particular explanation methods were evaluated using vector similarity measures to determine how well the label vectors resulting from this learned transform match the original label vectors. (A cross-validation protocol was employed to avoid over fitting.)

Figure 36 shows preliminary results suggesting that feature used in PRODIGAL can generate meaningful explanations for analysts, i.e. explanations that an analyst would give to another.

The study is described in more detail in [Goldberg 2016].

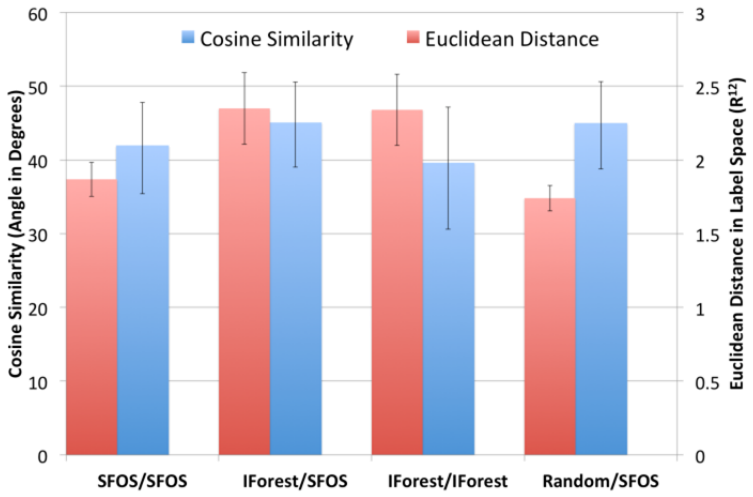


Figure 36. *Preliminary results suggesting that feature used in PRODIGAL can generate meaningful explanations for analysts*

Average similarity of labels transformed from Single Feature Outlier Scores (SFOS), and IForest - Independent Drop Out explanations. The second label is the type of explanation used to learn the transform.

System Development & Integration (Leidos SOW 1.2.2)

1.13. ADAMS Program Environment

Research work within the ADAMS research program required a large amount of data processing. This processing was conducted in an experimental environment where detection algorithms, source data, and operations were in flux based upon knowledge gained from previous activities. Each stage of research investigation required changes in processing, and the PRODIGAL system was developed to accomplish this with a minimum level of reconfiguration. A loosely coupled information system framework based upon the Spring Framework [Arthur 2005] provided a baseline data processing capability while simplifying the process of modifying the system as a whole.

PRODIGAL operates in the research environment as a manually controlled pipeline as illustrated in **Figure 37**. In this research environment, it was executed monthly to correspond with the red team scenario insertion and evaluation processes. As new versions of components were developed, tested, and considered for incorporation in the prototype, different versions and configurations modules were executed. For potential production environments, PRODIGAL can be configured to execute more frequently on different data periods (e.g., weekly execution on a rolling four-week period). This section describes the PRODIGAL components and the controlling software framework that enables this variety of different execution methods, beginning with the details of the components in the context of the monthly test bed processing that generated the majority of the results reported in this document.

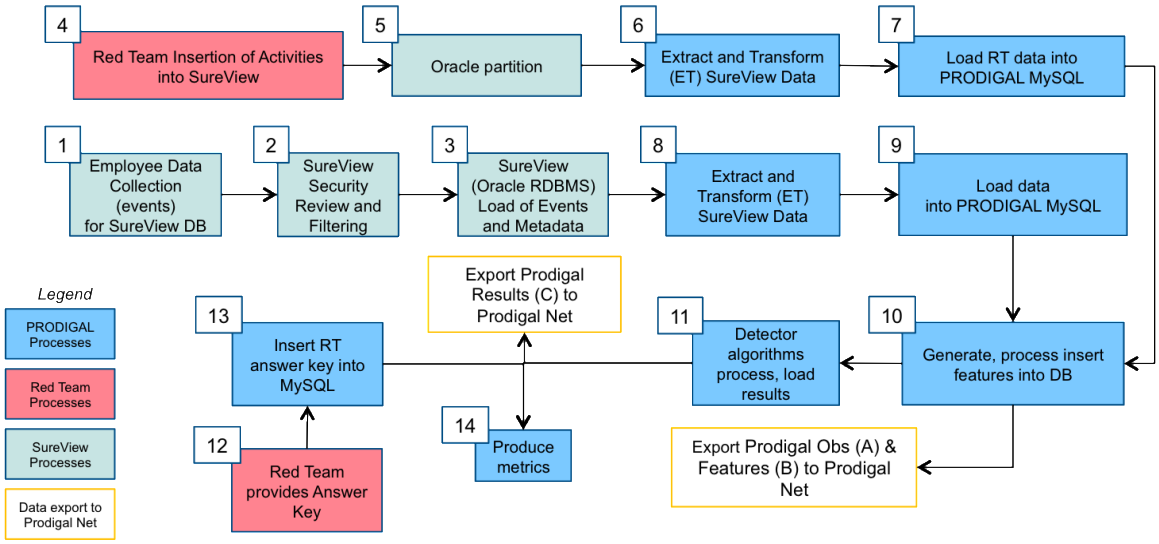


Figure 37. *Monthly PRODIGAL Pipeline Processing.*

Processing began with monitoring of events by SureView on user workstations and organizational servers. (boxes 1-3) Information collected by SureView was warehoused in an Oracle database at the SureView server and was used for regular monitoring by security personnel. A copy of the data was transmitted to the ADAMS test bed environment, where it was anonymized by removal or hashing of personally identifying information (PII) and stored in an instance of the SureView warehouse schema. In parallel, the Red Team created additional SureView events (boxes 4-5) that were inserted into a separate partition for merger with the collected data.

PRODIGAL processing began with an ETL component (box 8) that extracted data from the Oracle SureView database and (box 9) transformed and loaded it into the PRODIGAL schema in a MySQL database. The purpose of this transformation was to convert the data from SureView observations into user activities. The same ETL processes (boxes 6 and 7) were executed on the real user data and on the Red Team insert data.

The real and Red Team data were merged using table views, and PRODIGAL loaded this combined data into its MySQL database. This area of the PRODIGAL database is referred to as the PRODIGAL Observation Store. PRODIGAL next computed the features that served as the basis for its detectors and augmented the PRODIGAL Observation Store with these computed features (box 10).

Detectors, consisting of algorithms and their associated parameterizations, created anomaly scores for all user-days in the data set (box 11). Each detector separately scored each user-day, so there are many scores for each user-day. These scores are stored in a separate MySQL database called the PRODIGAL Results Store, and are indexed by algorithm id value, user id value, and a sequential run id value. The algorithm id specifies the exact algorithm used to generate a score. Algorithms read in many feature values from a user, a time period, or a population to assess how anomalous a behavior set is. The PRODIGAL prototype uses a user id numeric hash as a unique identifier for a specific person operating a computer.

The next step in the processing flow, also included in box 11, is the execution of the ensemble algorithm, which produces the official single score for each user-day. A user with a very high anomaly scores represents a candidate for further security investigation activities. The Red Team provided an answer key to the team after the pipeline was run (box 12). These labels are inserted into the database (box 13) and used to determine how accurate the output scores were and to compute the detection metrics discussed in this document (box 14).

1.14. ADAMS Environment Extensions

The ADAMS test bed environment interacted with two other environments as an extension of the research; 1) a research and development environment in which algorithm experimentation occurred on fully anonymized data and statistical summaries, and 2) an operational test environment in which the real data were processed (without the red team inserts) for evaluation by security personnel, with feedback on the highly-scored user-days provided back to the research team. Each environment was composed of multiple computers with different hardware configurations and source data sets.

Figure 38 shows a high level view of these environments and data sources. Arrows depict the flow of data from data sources to data recipients. The test bed environment contains all pipeline components shown in **Figure 37**. The research and development environment received data from the test bed environment subsequent to the processing step depicted in box 11 of **Figure 37**. The operational test environment received source data directly, with no anonymization, and executed the full pipeline. Researchers had no access to this environment.

Researchers used the research and development environment to create new framework and algorithm software. It is housed within a development network that is separate from all other data sources. It received data exports from the test bed environment, and acted as a development platform used by data analysts to explore results using novel and experimental processes. An engineering team used this to develop framework code to manage the overall PRODIGAL

prototype. Algorithm designers used this to experiment with algorithms, test approaches, and create production quality anomaly detection algorithms within a test environment.

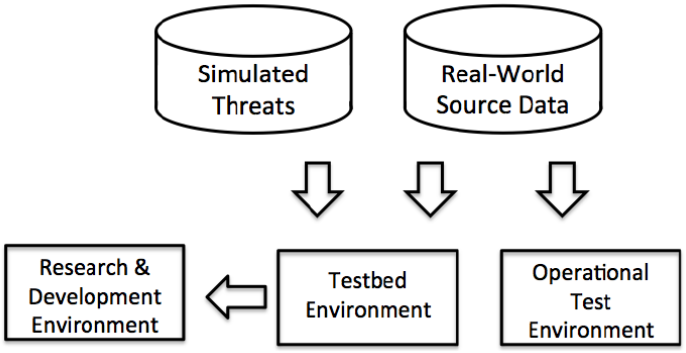


Figure 38. *Illustration of PRODIGAL R&D environment.*

1.15. Component Architecture

The PRODIGAL research framework allows researchers to create the components of the system separately. The PRODIGAL framework provided a computing system that connected components and orchestrated data processing stages between them. A set of configuration files were used by PRODIGAL to specify which components would execute during which phase of processing. These standardized configuration files, called *Flow* files, gave analysts the ability to rapidly swap out different ETL ("Overview of Extraction, Transformation, and Loading" 2014) processes, feature generation [Siddiqui 2015] calculations and machine learning [Jordan 2015] algorithm executions. The flexible nature of system definition mandated a mature configuration management process within the program, but also added the ability to approach a plug-and-play research environment.

The organization of PRODIGAL execution by way of Flow files provided a method to define the system data processing state. It also uses standard Spring dependency injection processes so the Flow files have some level of standardization beyond PRODIGAL. This simplifies learning how to execute the system components. System components did not reside in the same process space within PRODIGAL. Cross process communication was necessary to orchestrate the different concurrently executing processes based upon the configuration defined in the Flow file. PRODIGAL used a message transport layer based upon ActiveMQ [ActiveMQ 2016] to orchestrate the data processing flow.

The PRODIGAL framework ingests data that describe user activities on a computer. This user executes computer functions within a network of other users computers. Some of the users involved in activities are members of a single group of people (e.g. a workgroup), and some exist entirely outside of the purview of the PRODIGAL system and its stakeholders. The PRODIGAL data set is generated by a third party product, SureView™ [SureView 2010], which has been modified to protect user identity information. The ADAMS program collects data that describes user activities, and then further de-identifies PII data within that data set, exporting the results to JSON [JSON 2014] files which are indexed by event type, date, and time. **Figure 39** shows an example of this process.

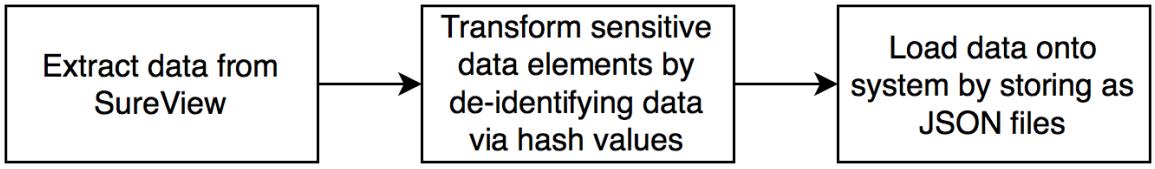


Figure 39. *ETL Process*

The feature generation reads in these event files and calculates numeric feature values based upon the content of these files. These feature values are generated based upon different temporal extents. Temporal extents refer to timespans of interest. For example, features may be calculated for each user based upon the events of a single day, or for that same user based upon the events within a week.

Feature values are stored within a MySQL [Oracle 2014a] relational database. These feature values are used by algorithm components to calculate scores. Each algorithm component uses the temporal based feature values to generate a score value that represents a degree of anomaly of this users behavior with their own past behavior, and the behavior of the general population. Algorithm calculations are stored within the results database as well. After all algorithms have been executed, an ensemble algorithm executes. The ensemble algorithm uses the results from all previously executed algorithms to assign an overall anomaly score to a users behavior.

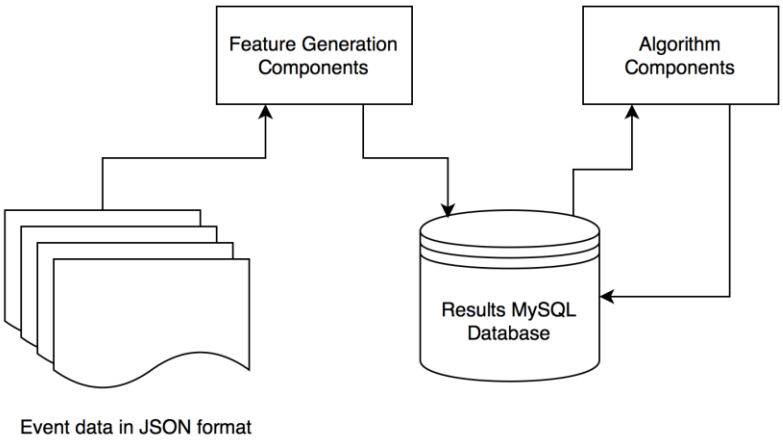


Figure 40. *Results database within system architecture.*

The ensemble-based scores allow system security specialists to rank user behaviors within the general population and across diverse time spans, and then select events of interest for investigation.

Many algorithms have the ability to generate explanation content as well. Explanation content describes what features provided a basis to developing an overall score. Algorithms seek to describe the anomalousness of a user's behavior. Explanations seek to describe why that score was generated.

The algorithms that generate explanations require a significant increase in run time. To mitigate this effect, only a subset of all user days generates explanation output. This subset is based on a percentage of top-scoring ensemble scores. PRODIGAL will typically create an explanation file for users scoring in the top 10% scoring population members.

1.16. Executing PRODIGAL

PRODIGAL data processing is executed via command shell. Each type of data processing (ETL, Feature Generation, Algorithms, or Metrics) is triggered by running a command to trigger a Spring Framework dependency injection process [Yang 2008]. Two java applications provide an ability to read in a configuration XML flow file (shown in **Figure 41**) and dynamically load in a diverse set of Java software classes. Some of these classes will execute non-java applications through automated shell commands.

```
<?xml version="1.0" encoding="UTF-8"?>
<Flow id="2314" name="OSU1AlUdCoUqCROSS" xmlns="http://prodigal.saic.com/flow"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://prodigal.saic.com/flow ../../main/xsd/flow.xsd "
temporalStartDate="03-01-2013"
    temporalEndDate="03-31-2013" temporalType="6">
    <wrapper>
        <name>DTAllUdCoUvq</name>
                <algorithm>
            <precedence>
                <creator>TemporalRange</creator>
            </precedence>
        </algorithm>
    </wrapper>
   <wrapper isTerminator="true">
        <name>CrossPrediction</name>
                <algorithm>
            <precedence>
                <creator>DTAllUdCoUvq</creator>
            </precedence>
        </algorithm>
    </wrapper>
```

Figure 41. *Example configuration XML flow file*

The application WrapperExecutor.jar is used to process a flow file and dynamically create software that may access the MySQL results database. The DBPrepExecutor.jar file is used within the Vegas experiment environment to access the Oracle SureView database, and transform that data into the JSON data. Each of the flow files will have the same type of format.

The flow file is defined by a hierarchy with a top level called Flow. The Flow section contains a set of names that correspond to configuration files. Those configuration files are parameterized to ensure that the correct inputs to different processes are made, even when they share common software components. Each of the software components is contained in a section named wrapper. The wrapper software execution loads each wrapper item, and ensures that a proper precedence order is kept.

1.17. Transitioning PRODIGAL to a real-world enterprise

Prodigal is designed as a research platform. When needed, engineers may transition PRODIGAL into a real-world environment. This section discusses the different tasks that may be required prior to performing a successful transition. The list below summarizes the updates needed to deploy into an environment that does not match the existing PRODIGAL environment.

1. Aquire data – this usually requires a method of collecting different data elements from users as they perform their normal daily functions. PRODIGAL uses SureView to perform this function. Not all environments will have access to SureView. SureView may be configured to collect data differently than the experimental environment as well.

A transition environment will need to export its data set to a the JSON format that PRODIGAL ingests. This format contains the date, hour, data type, and an array of all events for that data type within a 1-hour time frame. Stakeholders must export their data to a directory to allow for feature generation.

2. Generate features – In the event that new data types or content exists, then software must read in the raw JSON event files and create feature values. Feature values are normally calculated by counting events or values with the period, finding maximums or minimums across the time period, or by performing statistics within a time period. These feature values must be given a unique integer ID value then stored in the results database. Each of the feature generation processes that requires new software should be added to a flow file.
3. Calculate Scores – The algorithms supplied with PRODIGAL can be quickly reconfigured. This is necessary when new feature types have been added to the system. Transition teams must update the Spring framework dependency injection resources if new feature types have been added. If new algorithms have been added to the system, then those algorithms must be paired with a new Spring framework dependency injection resource.

PRODIGAL is built in Java, and installed using a Redhat Package Manager (RPM) [RPM 2014] software package. Additional algorithms may be installed as well. This is done as a simple file copy process.

PRODIGAL is run in the experimental environment as a large batch process that reads in an entire data set and then processes it. Real world operations may have time constraints that preclude this. In a real world deployment, PRODIGAL data processing should be executed daily or hourly.

Supporting the End User (Leidos SOW 2.2.1)

1.18. The Analyst Interface

The PRODIGAL analyst interface focuses on the data most likely to contain true anomalies that are indicative of insider threat activity and providing a starting point for investigation. The AI presents analysts with a list of users that is ranked by PRODIGAL's ensemble algorithm. Analysts are able to select a user, compare the user's behavior for a given day compared to all users for all features, and then drill down into specific activities by data type. If the analyst finds any of these behaviors of interest, he or she can then explore those behaviors further in SureView.

1.18.1. Presenting Ensemble Scores in Context

This section presents examples of the use of single feature outlier scores in the PRODIGAL Analyst Interface (AI). Analysis starts with a list of entity extents (user-days) sorted by highest ensemble anomaly score. The analyst compares these scores with others for the date or the entire month using the display shown in **Figure 42**. The AI presents user day scores using a box plot, with the upper whisker representing the top 5 percent of scores in order to highlight the most anomalous behaviors. The black squares represent the selected user's scores for every day. User 410400 has scores that are in the top 5 % for several days in the month. The day highlighted by the green box is the highest-ranked day for the entire month.

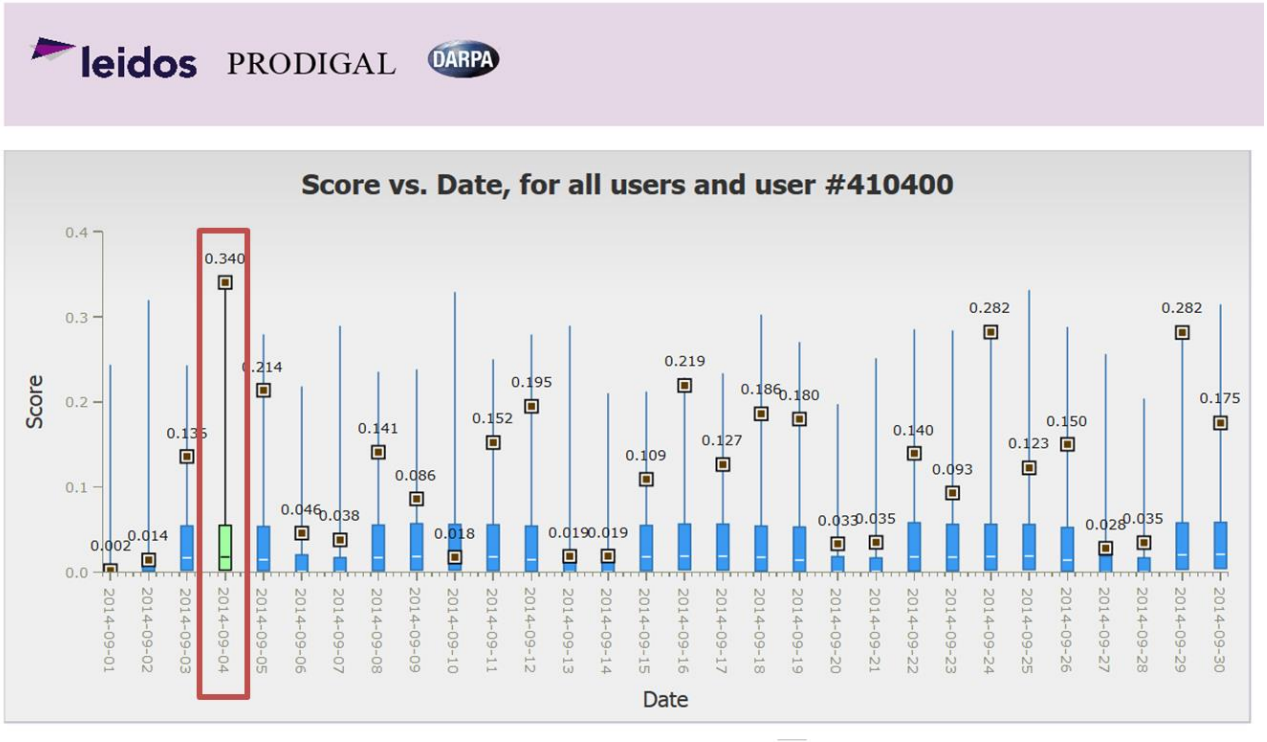


Figure 42. *User #410400's scores compared to the baseline population for the month. The black squares represent the user's scores. The red box highlights the day in question for the user, which is September 9, 2014.*

For a given day of interest, the AI enables the analyst to view the individual features associated with the ensemble score, allowing the analyst to focus on specific anomalous behaviors while investigating a particular scored entity, as shown in **Figure 43**. The AI lists the data type

associated with the feature score (e.g., file, email, URL, printer, logon), a summary description of the feature name, and the normalized score for that feature.

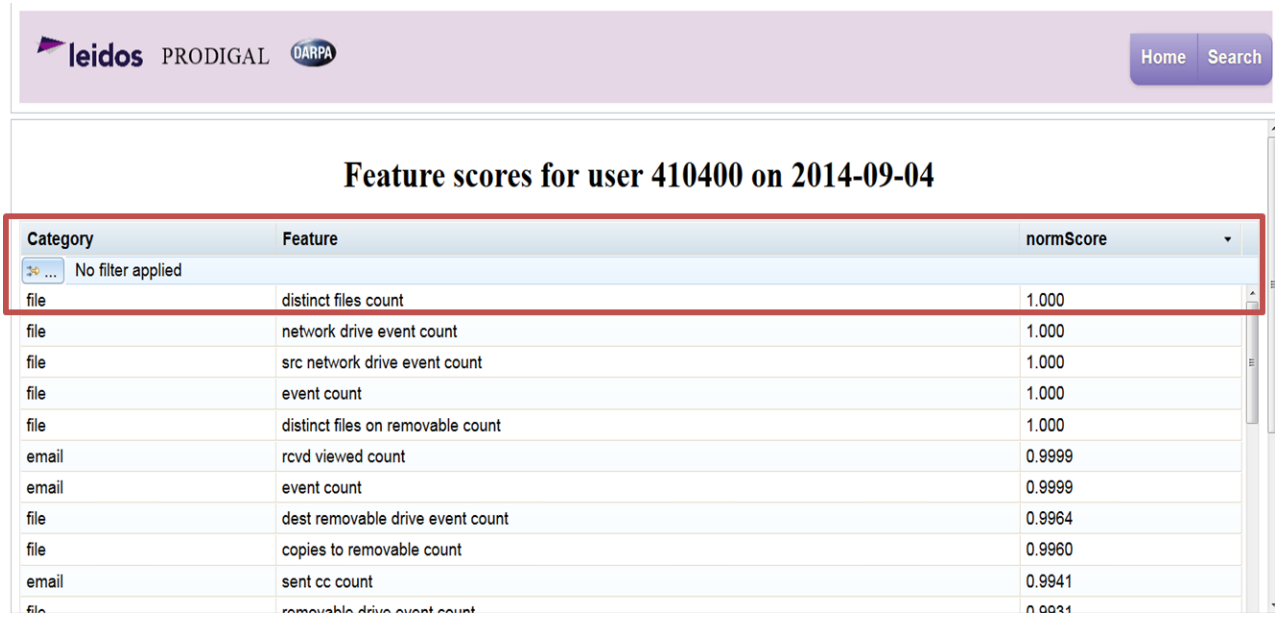


Figure 43. *The individual feature scores for user 410400 on September 9, 2014. The user is in the less than 0.0001% percentile of all users on this day with respect to multiple file features.*

Finally, drill-down to underlying user-computer transactions is included to let the analyst view the behavior from which features and ultimately anomaly scores were computed. (**Figure 44**) (Note, to preserve privacy in the research database, numerical hash keys replace unique user names, file names, domain names, and email addresses. A live implementation of PRODIGAL would present these to the analyst.) Inclusion of the underlying observations associated with the feature scores enables the analyst to visually inspect the data and assess whether the user's unusual behavior is concerning and merits further exploration (outside of PRODIGAL).

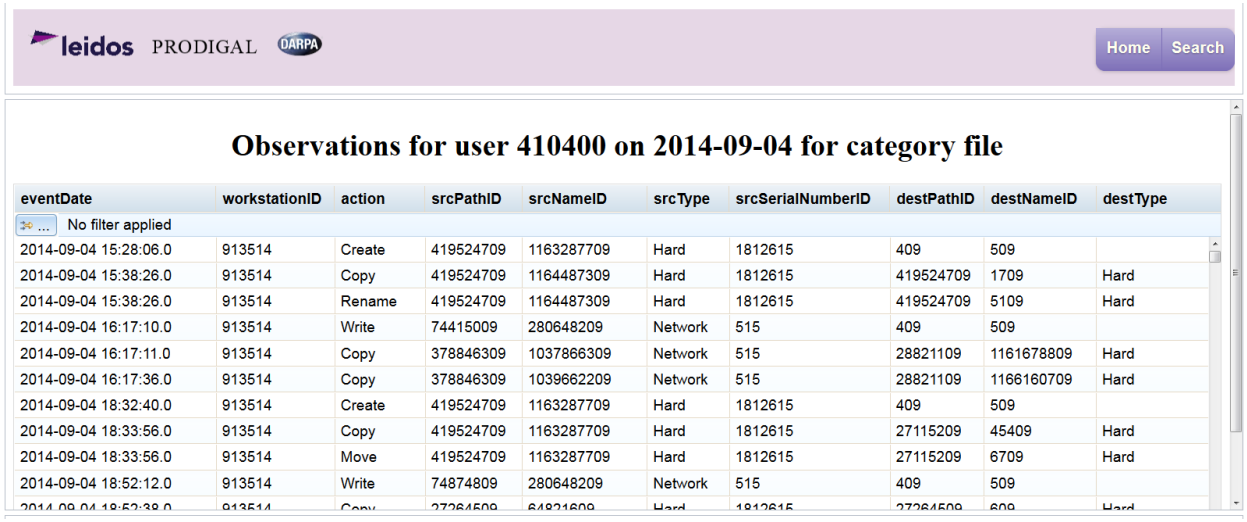


Figure 44. *Individual observations and select attributes associated with the top-ranked individual feature score, "Distinct Files Count".*

1.19. Explaining Anomaly Scores

Level 0 Explanations were integrated into the Analyst's Interface. The AI starts with a list of entity extents (user-days) sorted by highest ensemble anomaly score. The analyst may compare these user scores with others for the date or the entire month using the display shown in **Figure 45**.

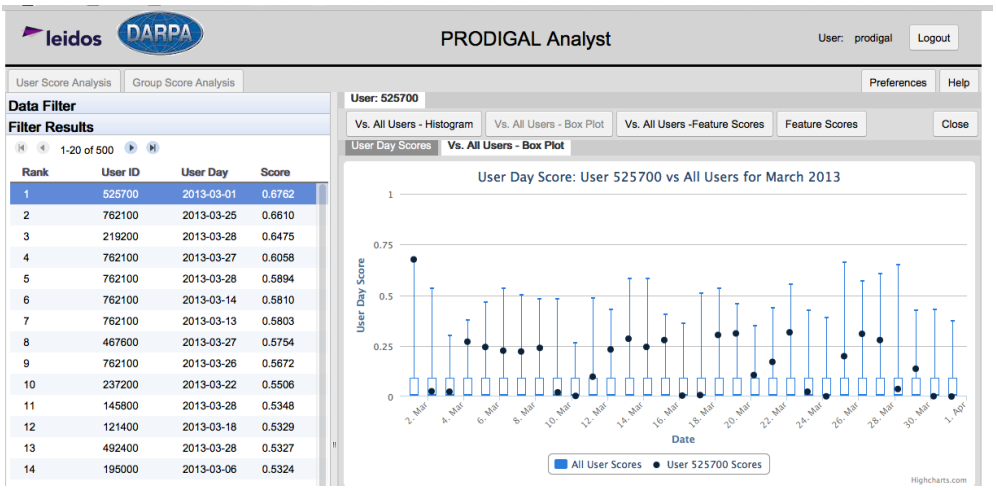


Figure 45. *Overall user-day scores compared to all users over the month.*

The list of explanations is presented to the analyst, sorted by outlier score (not shown). Selecting the features with highest outlier score further allows the analyst to focus on the most anomalous behavior while investigating a particular scored entity.

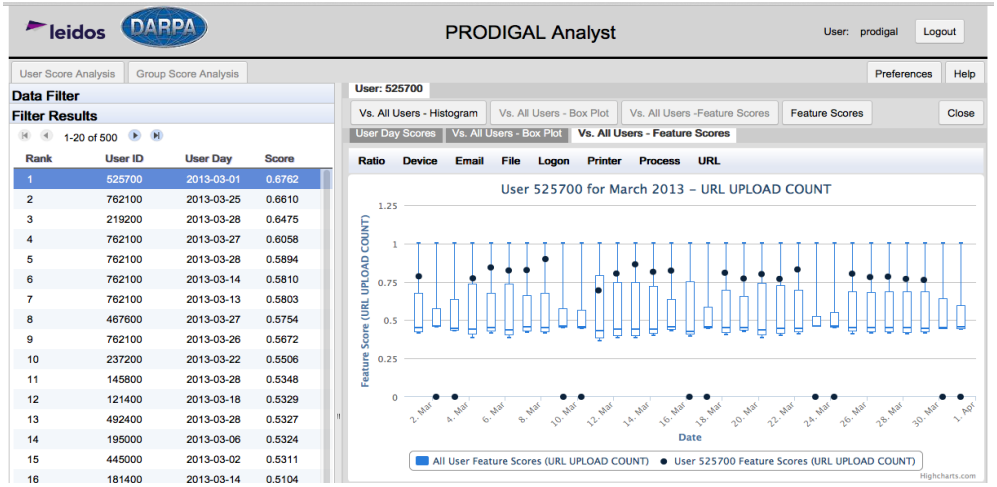


Figure 46. *URL upload outlier scores compared to all users over the month.*

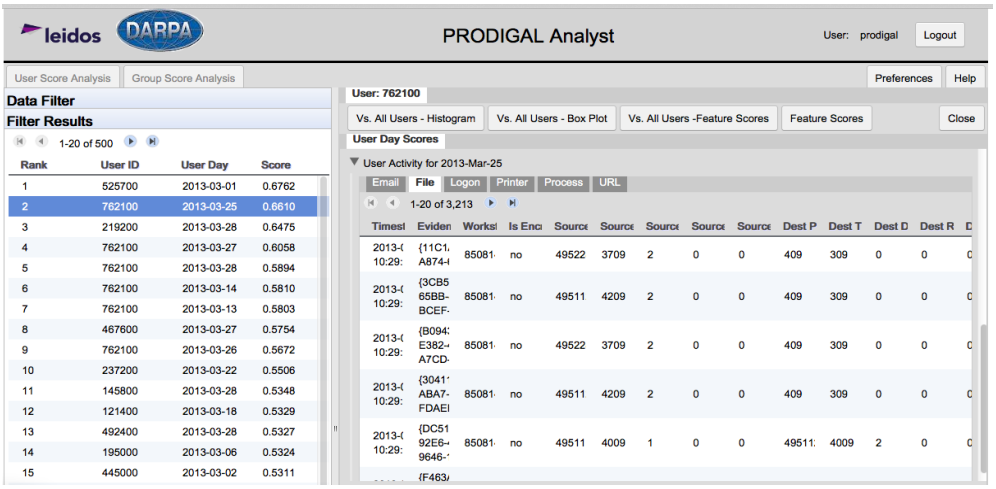


Figure 47. *Detailed drill-down of file access events.*

Finally, drill-down to underlying user-computer transactions is included to let the analyst view the behavior from which features and ultimately anomaly scores were computed. (**Figure 47**) *(Note, to preserve privacy in the research database, numerical keys replace unique usernames, filenames, domains, and email addresses. A live implementation of PRODIGAL would present these to the analyst in clear text.)*

Level 1 explanations, when computed can also be displayed by the AI, however this is not a feature of the standard PRODIGAL release.

Transition Preparation and Recommendations for Future Operational Evaluation Efforts (Leidos SOW 2.2.3)

1.20. Operational Evaluation Framework.

Despite the promising results, the DARPA evaluation did not allow for the performers to investigate specific beyond the Red Team inserts due to the nature of the ADAMS program data. **An operational evaluation will provide transition partners an opportunity to assess the suitability and utility of PRODIGAL as the back-end machine-learning, triage analytics system in the context of insider threat detection.** Evaluation of PRODIGAL's capabilities in the context of operational workflows also facilitates exploration of concepts of operations for integrating analytics with user activity monitoring data, as well as opportunities for incorporating new data types (e.g., badge-in/badge-out data).

1.20.1. Task 1: Installation and Configuration.

The installation and configuration of PRODIGAL occurs at a facility as designated by the sponsoring organization, especially one that already uses InnerView for its user activity monitoring capability. The major components are database; the extract, transform, and load (ETL) pipeline; PRODIGAL analytics – a software suite that includes anomaly detectors (over 100 in total), algorithms that provide explanations of detector results, and the ensemble algorithm; and the front-end, web-based Analyst Interface. PRODIGAL hardware requirements is dependent on size of the evaluation user base, with a Red Hat 64 operating system (OS) with 16 cores, 128 GB of RAM, and at least 8 TB or hard drive space required for a 5500 user population. Software requirements include Java 1.7, JMS, MySQL, Apache Tomcat 8, R, python and various python packages, and Lisp. Leidos has documentation to support installation, and the PRODIGAL team would work with the sponsor's engineering and security personnel to identify and resolve information assurance issues related to installation and configuration tasks.

1.20.2. Task 2: Extract and Process Features.

Leidos engineers assess the sponsor's user activity monitoring data, characterizing the data types and attributes collected prior to ingest by PRODIGAL. PRODIGAL supports features specified from count-based observations (e.g., the number of logon events by user, the number of email sent by a given user). Feature design incorporated domain knowledge provided by subject matter experts and are intended to reflect behaviors of relevance to known and unknown but plausible insider threat scenarios. Features consist of an entity extent (e.g., user; a file; a workstation ID; and a temporal extent. The default temporal extent in PRODIGAL is a user day, which is defined as a 24-hour period (i.e., not a business day). Currently, there are 111 features in PRODIGAL based on email, file, groups (based on LDAP information and shared entities such as printers, files, and workstations), logon, printer, process, and URL data types. PRODIGAL also has over 20 ratio features that compare activities within and across data types (e.g., the ratio of the number of all URL events to the number of URL upload events. **Table 14** below summarizes the features already specified in PRODIGAL.

Table 14. *Feature Summary for Operational Evaluation of PRODIGAL.*

Observation Type	Number of Features	Examples
Email	18	Count of attachments on sent emails
File	28	Count of file events to removable drives

Group	11	Shared printers
Login	4	Count of distinct workstations logged onto
Printer	9	Count of print jobs submitted
URL	13	Count of Blacklist events
Ratio	28	Ratio of file events on removable drives to all file events Ratio of URL uploads to URL downloads

Leidos engineers configure PRODIGAL's ETL pipeline to process observations in either batch/stand-alone mode or as a feed from an instance of IV. As part of the feature extraction process, Leidos data scientists perform exploratory data analysis to identify meaningful peer groups, look-back periods, and distributions of activities involving monitored information technology resources, which are the necessary parameters for baselining user behaviors. Leidos data scientists and engineers also test and assess the quality of the feature extraction prior to processing by analytics, comparing ETL pipeline processing times, throughput, and output in the operational environment to results achieved in lab or other experimental settings.

1.20.3. **Task 3: Run analytics, present results to analysts, assess findings.**

PRODIGAL's anomaly detectors run on the features extracted from user activity monitoring data, providing inputs to the ensemble algorithm, which in turns fuses the detector output into a single score. The single score indicates the unusualness of a given user compared to all users on any given day in a time period. PRODIGAL analytics also provide feature-based explanations to the analyst, providing additional context to support the interpretation of results in domain terms. The PRODIGAL Analyst Interface (AI) is the system's front-end component that exposes the analyst to both the ensemble output and feature-based explanations. While PRODIGAL analytics are automated, Leidos data scientists configure the detectors and the ensemble and explanation algorithms and assess the quality and performance of the output. Leidos data scientists with support from university researchers also adjust PRODIGAL analytics to previously un-encountered but suspected simple or complex behaviors (e.g., malicious insider activity involving a single data type or threaded activities involving multiple data types).

1.20.4. **Task 4: Collect, compute, and interpret metrics.**

Task 4 goal is to demonstrate PRODIGAL AI capabilities and functionality to analysts at the host agency. Operational evaluation performers will assist with the integration of PRODIGAL analytics (the components and the information presented to analysts) into the host agency's workflows and procedures. Candidate metrics entail utility and effectiveness. We propose two measures of utility. The first is relevance, which we define as a ratio of the number of user days that an analyst labels as being actually unusual to a cutoff point in the ranked list of PRODIGAL results as determined by the analyst (e.g., top 10, top 20, top 50 user days). The second measure of utility is the error rate, which is a ratio of the number of user days that the analyst labels as not unusual or otherwise of interest to a cutoff point in the ranked list of PRODIGAL results. Effectiveness include time-based measures such as the time needed to determine whether a user's behaviors on a given individual user days is either unusual, but explainable, or unusual and meriting further investigation.

1.20.5. **Proposed Operational Evaluation Schedule.**

Table 15 presents nominal milestones and associated deliverables for the project, along with the delivery date based on a 4 FTE team of engineers, data scientists, and insider threat researchers

with university researcher support. Deliverable types include a demonstration (D), product (P) or report (R), where a report may constitute a document, presentation, or spreadsheet, following on the sponsor's preference.

Table 14. *Nominal PRODIGAL Operational Evaluation Schedule.*

Milestone (deliverable type)	**Date**
Initial integration with local SV/IV data for scale of 1K users (D, P). Initial data schema map and feature definition. (P)	Month 4
Hardening, scaling of PRODIGAL to meet user demand (5K users) (D). Completion of new feature generation (P).	Month 8
Results and findings from user-based evaluations (P, R)	Months 10 through 16
Report progress against Red Team inserts (optional)	Months 12 though 16
Status reports (R), final report (R)	Monthly, Month 12, 18

References

[ActiveMQ 2016] Apache Software Foundation. 2016. “Apache ActiveMQ.” ActiveMQ. March 7. http://activemq.apache.org/.

[Ankerst 1999] Ankerst, Mihael, et al. "OPTICS: ordering points to identify the clustering structure." *ACM SIGMOD Record* 28.2 (1999): 49-60.

[Akoglu 2012] Akoglu, L., Chau, D. H., Kang, U., Koutra, D., and Faloutsos, C. (2012). Large Graph Mining System for Patterns, Anomalies & Visualization. 16th Pacific-Asia Conference, PAKDD 2012. Kuala Lumpur, Malaysia.

[Akoglu 2013a] Akoglu, L., Chandy, R., and Faloutsos, C. (2013a). Opinion Fraud Detection in Online Reviews using Network Effects. *Proceedings of the 7th Annual* International *Conference on Weblogs and Social Media.* Cambridge, Massachusetts, USA. Association for the Advancement of Artificial Intelligence.

[Akoglu 2013b] Akoglu, L., Vreeken, J., Tong, H., Chau, D.H., and Tatti, N.(2013b). Mining Connection Pathways for Marked Nodes in Large Graphs. *Proceedings of the Society for Industrial and Applied Mathematics 2013 SIAM Conference on Data Mining (SDM13).* Austin, Texas, USA.

[Araujo 2014] Araujo, M., Papadimitriou, S., Günnemann, S., Faloutsos, S., Basu, P., Swami, A., Papalexakis , E., and Koutra, D., (2014) Com2: Fast Automatic Discovery of Temporal ('Comet') Communities Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Tainan, Taiwan, May 13-16 2014

[Arbour 2016] D. Arbor and D. Jensen (2016). Learning with mixtures of dependency networks. Under review.

[Arthur 2005] Arthur, J., and S. Azadegan. 2005. “Spring Framework for Rapid Open Source J2EE Web Application Development: A Case Study.” In , 90–95. IEEE. doi:10.1109/SNPD-SAWN.2005.74.

[Berlingerio 2012a] Berlingerio, M., Koutra, D., Eliassi-Rad, T., and Faloutsos, C. (2012a). NetSimile: A Scalable Approach to Size-Independent Network Similarity. Paper presented at the Workshop on Information in Networks 2012.

[Bettadapura 2015] V. Bettadapura, E. Thomaz, A. Parnami, G. Abowd, and I. Essa, “Leveraging Context to Support Automated Food Recognition in Restaurants,” in Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV), 2015. (http://www.cc.gatech.edu/~irfan/p/2015-Bettadapura-LCSAFRR.pdf)

[Brandes 2001] U. Brandes, "A faster algorithm for betweenness centrality," Journal of Mathematical Sociology, vol. 25, no. 2, pp. 163-177, 2001.

[Breunig 1999] Breunig, Markus M., et al. "Optics-of: Identifying local outliers." *Principles of data mining and knowledge discovery*. Springer Berlin Heidelberg, 1999. 262-270.

[Cappelli 2012] D. Cappelli, A. Moore, R. Trzeciak, The CERT Guide to Insider Threats: How to Detect, Prevent, and Respond to Information Technology Crimes. Addison-Wesley Professional. 2012.

[Chau 2012a] Chau, D. H. (2012a). Data Mining Meets HCI: Making Sense of Large Graphs. (Doctoral dissertation). Carnegie Mellon University, School of Computer Science. Pittsburgh, Pennsylvania, USA. (Dissertation Award Honorable Mention).

[Chau 2014] Chau, D. H., & Faloutsos, C. (2014). Case study on fraud detection using social-network analysis. In Ahaji, R. & Rokne, J. (Eds). Encyclopedia of Social Networks and Mining. Springer.

[Crammer 2006] Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., & Singer, Y. (2006). Online Passive-Aggressive Algorithms. *Journal of Machine Learning Research, 7*, 551–585.

[Deng 2014] Deng, K., Han, S., Li, K. J., & Liu, J. S. (2014). Bayesian Aggregation of Order-Based Rank Data. *Journal of the American Statistical Association, 109* (507), 1023–1039.

[Dietterich 2000] T. Dietterich. Ensemble Methods in Machine Learning. In Multiple Classifier Systems, 115. Springer, 2000.

[Dojo 2016] The Dojo Foundation. 2016. "The Dojo Toolkit." https://dojotoolkit.org/.

[Ediger 2012] D. Ediger, R. McColl, J. Riedy and D.A. Bader, "STINGER: High Performance Data Structure for Streaming Graphs," Conference Presentation, The 16th Annual High Performance Embedded Computing Workshop (HPEC), Lexington, MA, September 10-12, 2012.

[Emmott et. al., 2013] Andrew Emmott, Shubhomoy Das, Thomas Dietterich, Alan Fern and Weng-Keen Wong. (2013). Systematic Construction of Anomaly Detection Benchmarks from Real Data. SIGKDD Workshop on Outlier Detection and Description.

[Figueiredo 2014] F. Figueiredo, J. M. Almeida, Y. Matsubara, B. Ribeiro, and C. Faloutsos. (2014). Revisit behavior in social media: The Phoenix-R Model and Discoveries. In ECML/PKDD.

[Friedland 2014] L. Friedland, A. Gentzel, and D. Jensen (2014) Classifier-adjusted density estimation for anomaly detection and one-class classification. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, Philadelphia, PA. pp. 578-586.

[Gatterbauer 2014] Gatterbauer, W., Guennemann, S., Koutra, D., Faloutsos, C., (2014). Linearized and Single-Pass Belief Propagation. Proceedings of the VLDB Endowment, Volume 8(4) (VLDB'15)

[Goldberg 2016] Goldberg, H. G., Young, W. T., Memory, A. C., Senator, T.E., "Explaining and Aggregating Anomalies to Detect Insider Threats", 49th Hawaii International Conference on System Sciences, Pp. 2739-2748, IEEE Computer Society, 2016.

[Goldberg 2017] Goldberg, H. G., Young, W. T., Reardon, Matthew G., Phillips, Brian J., Senator, T.E., "Insider Threat Detection in PRODIGAL", 50th Hawaii International Conference on System Sciences, IEEE Computer Society, 2017 (pending).

[Green 2012] O. Green, R. McColl, D. Bader, "A Fast Algorithm For Streaming Betweenness Centrality", 4th ASE/IEEE International Conference on Social Computing, 2012.

[Green 2013] O. Green, D. Bader, "Faster Betweenness Centrality Based on Data Structure Experimentation", 13th International Conference on Computational Science (ICCS), 2013.

[Günnemann 2014] Günnemann, N., Günnemann ., N. and Faloutsos, C., (2014) Robust Multivariate Autoregression for Anomaly Detection in Dynamic Product Ratings, International World Wide Web Conference (WWW), 2014

[Hastie 2001] T. Hastie, R. Tibshirani, and J. H. Friedman (2001). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. New York: Springer-Verlag.

[Heckerman 2001] D. Heckerman, M. Chickering, C. Meek, R. Rounthwaite, & C. Kadie (2001). Dependency networks for inference, collaborative filtering, and data visualization. *The Journal of Machine Learning Research*, *1*, 49-75.

[Jiang 2014] Jiang, M., Cui, P., Beutel, A., Faloutsos C.and Yang S. (2014). CatchSync: Catching Synchronized Behavior in Large Directed Graphs, KDD 2014, New York City, NY, USA, Aug. 24-27.

[Jordan 2015] Jordan, M. I., and T. M. Mitchell. 2015. "Machine Learning: Trends, Perspectives, and Prospects." Science 349 (6245): 255–60. doi:10.1126/science.aaa8415.

[JSON 2014] json.org. 2014. "JSON." Accessed July 10. http://www.json.org/.

[Kang 2014] U Kang, Jay-Yoon Lee , Danai Koutra and Christos Faloutsos (2014) Net-Ray: Visualizing and Mining Billion-Scale Graphs Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), 2014, Tainan, Taiwan, May 2014

[Kang 2011] Kang, U., Chau, D. H., & Faloutsos, C. (2011, April). Mining large graphs: Algorithms, inference, and discoveries. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on* (pp. 243-254). IEEE.

[Kriegel 2008] Kriegel, H.-P., Schubert, M., & Zimek, A. (2008). Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)* (pp. 444–452). http://doi.org/10.1145/1401890.1401946Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2012). Isolation-Based Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data*, *6*(1), 1–39.

[Koutra 2012] Koutra, D., Papalexakis, E. E., Faloutsos, C. (2012). TENSORSPLAT: Spotting Latent Anomalies in Time. *Proceedings of PCI '12, the 16th PanHellenic COnfernece on Informatics with International Participation*. Thessaloniki, Greece. IEEE Comptuer Society, Washington, DC, USA.

[Koutra 2013a] Koutra, D., Koutras, V., Prakash, B.A., and Faloutsos, C. (2013a). Patterns among Competing Tasks Frequencies: Super-Linearities, and the Almond-DG Model. In J. Pei, V.S. Tseng, H. Motoda, and G. Xu, *Advances in Knowledge Discovery and Data Mining: 17th Pacific-Asia Conference, PAKKDD 2013, Gold Coast Australia, April 14-17 2013Proceedings Part 1.* Gold Coast, Australia. Springer Lecture Notes in Computer Science.

[Koutra 2013b] Koutra, D., Vogelstein, J.T., Faloutsos, C. (2013b.) DELTACON: A Principled Massive-Graph Similarity Function. *Proceedings of the Society for Industrial and Applied Mathematics 2013 SIAM Conference on Data Mining (SDM13).* Austin, Texas, USA.

[Koutra 2014] Koutra, D., Kang, U., Vreeken, J., & Faloutsos, C. , (2014). VOG: Summarizing and Understanding Large Graphs. SDM 2014, Philadelphia, PA, April 2014.

[Lazarevic 2005] A. Lazarevic and V. Kumar. Feature Bagging for Outlier Detection. In Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 157166, 2005.

[Lee 2012] Lee, J.-Y., Kang, U., Koutra, D., and Faloutsos, C. (2012). Fast anomaly detection despite the duplicates. *Computer Science Technical Reports 2012,* CMU-CS-12-146. School of Computer Science, Carnegie Mellon University. Pittsburgh, Pennsylvania, USA.

[Lee 2013] Lee, J.-Y., Kang, U., Koutra, D., and Faloutsos, C. (2013). Fast anomaly detection despite the duplicates. *Proceedings of the 22nd International Conference on World Wide Web Companion.* Brazil.

[Lin 2013] Zhiyuan (Jerry) Lin, Duen Horng (Polo) Chau. Interactive Multi-resolution Exploration of Million Node Graphs. *Poster Abstract, IEEE VIS 2013* Oct 13 - 18, 2013. Atlanta, GA, USA.

[Lin 2013a] Zhiyuan (Jerry) Lin, Nan Cao, Hanghang Tong, Fei Wang, U Kang, and Duen Horng (Polo) Chau. Demonstrating Interactive Multi-resolution Large Graph Exploration. *ICDM 2013*. Dec 7 - 10, 2013. Dallas, Texas, USA.

[Liu 2008] Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008, December). Isolation forest. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on* (pp. 413-422). IEEE.

[Memory 2013] Memory, A., Goldberg, H. G., & Senator, T. E. (2013, June). Context-aware insider threat detection. In *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*.

[Memory 2015] Memory, A., Senator, T. (2015). *Towards Robust Anomaly Detection Ensembles using Explanations*. KDD 2015 Workshop on Outlier Definition, Detection, and Description.

[Neville 2007] J. Neville & D. Jensen (2007). Relational dependency networks. *The Journal of Machine Learning Research*, *8*, 653-692.

[Oracle 2014a] Oracle. 2014a. "MySQL :: The World's Most Popular Open Source Database." Mysql.org. http://www.mysql.com/.

[Oracle 2014b] "Packaging Web Archives." https://docs.oracle.com/javaee/7/tutorial/packaging003.htm.

[Oracle 2014c] "Overview of Extraction, Transformation, and Loading." 2014. Accessed September 2. http://docs.oracle.com/cd/B19306_01/server.102/b14223/ettover.htm.

[Papalexakis 2013] Papalexakis, E.E., Akoglu, L., and Ienco, D. (2013). Do more Views of a Graph help? Community Detection and Clustering in Multi-Graphs. *Proceedings of the 1thth International Conference on Information* Fusion. Istanbul, Turkey.

[Pevný 2016] Pevný, T. (2016). Loda: Lightweight on-line detector of anomalies. *Machine Learning*, *102*(2), 275-304.

[Pienta 2014] Robert Pienta, Acar Tamersoy, Hanghang Tong, and Duen Horng Chau. MAGE: Matching Approximate Patterns in Richly-Attributed Graphs. *Proceedings of IEEE BigData 2014 conference.* Oct 27-30, Washington DC, USA.

[Pienta 2015a] Robert Pienta, James Abello, Minsuk Kahng, Duen Horng Chau. Scalable Graph Exploration and Visualization: Sensemaking Challenges and Opportunities. *International Conference on Big Data and Smart Computing (BigComp)*. Jeju Island, Korea. February 9-12, 2015.

[Pienta 2015b] Robert Pienta, Acar Tamersoy, Hanghang Tong, Alex Endert, Duen Horng (Polo) Chau. Interactive Querying over Large Network Data: Scalability, Visualization, and Interaction Design. *ACM Conference on Intelligent User Interfaces (IUI)*. Atlanta, GA, USA. March 29 - April 1, 2015.

[Plackett 1975] Plackett, R. L. (1975). The Analysis of Permutations. *Applied Statistics*, *24*(2), 192–202.

[RPM 2014] "Rpm Package Manager." 2014. Accessed September 2. http://rpm5.org/.

[Schubert 2012] Schubert, E., Zimek, A., & Kriegel, H.-P. (2012). On Evaluation of Outlier Rankings and Outlier Scores. In *12th SIAM International Conference on Data Mining (SDM)* (pp. 1047–1058).

[Senator 2013] Senator, T., & et al. (2013). Detecting Insider Threats in a Real Corporate Database of Computer Usage Activity. ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) Chicago, Illinois, USA.

[Sharma 2015] Y. Sharma, V. Bettadapura, T. Ploetz, N. Hammerla, S. Mellor, R. McNaney, P. Olivier, S. Deshmukh, A. Mccaskie, and I. Essa, "Video Based Assessment of OSATS Using Sequential Motion Textures," in Proceedings of Workshop on Modeling and Monitoring of Computer Assisted Interventions (M2CAI), 2014. (http://www.cc.gatech.edu/~irfan/p/2014- Sharma-VBAOUSMT.pdf)

[Sharma 2014] Y. Sharma, T. Ploetz, N. Hammerla, S. Mellor, R. McNaney, P. Oliver, S. Deshmukh, A. McCaskie, and I. Essa, "Automated Surgical OSATS Prediction from Videos," in Proceedings of IEEE International Symposium on Biomedical Imaging, Beijing, CHINA, 2014. (http://www.cc.gatech.edu/~irfan/p/2014-Sharma-ASOPFV.pdf)

[Siddiqui 2015] Siddiqui, A., Fern, A., Dietterich, T. G., & Wong, W. (2015). *Sequential Feature Explanations for Anomaly Detection*. KDD 2015 Workshop on Outlier Definition, Detection, and Description. arXiv:1503.00038v1

[Stolper 2014a] Charles D. Stolper, Minsuk Kahng, Zhiyuan Lin, Florian Foerster, Aakash Goel, John Stasko, and Duen Horng Chau. GLO-STIX: Graph-Level Operations for Specifying Techniques and Interactive eXploration. *IEEE InfoVis 2014, November 9-14, Paris, France.*

[Stolper 2014b] Charles D. Stolper, Florian Foerster, Minsuk Kahng, Zhiyuan Lin, Aakash Goel, John Stasko, Duen Horng (Polo) Chau. GLOs: Graph-Level Operations for Exploratory Network Visualization. *Extended Abstracts, CHI 2014*. Apr 26 - May 1, 2014. Toronto, Canada.

[SureView 2010] Raytheon Corporation. 2010. “SureViewTM Proactive Endpoint Information Protection.” http://www.raytheon.com/capabilities/rtnwcm/groups/iis/documents/content/rtn_iis_sureview_datasheet.pdf.

[SureView 2013] SureView Proactive Endpoint Information Protection, Raytheon, February 13, 2013.

[Wallnau 2014] K. Wallnau, et. al. ”Simulating malicious insiders in real host-monitored user data.” In Proceedings of the 7th USENIX conference on Cyber Security Experimentation and Test, pp. 4-4. USENIX Association, 2014.

[Yang 2008] Yang, Hong Yul, Ewan Tempero, and Hayden Melton. 2008. “An Empirical Study into Use of Dependency Injection in Java.” In , 239–47. IEEE. doi:10.1109/ASWEC.2008.4483212.

[Yibin 2014] Yibin Lin, Agha Ali Raza, Jay-Yoon Lee, Danai Koutra, Roni Rosenfeld , Christos Faloutsos (2014), Influence Propagation: Patterns, Model and a Case Study, Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), 2014, Tainan, Taiwan, May 13-16 2014

[Young 2013] W T. Young et. al, Use of Domain Knowledge to Detect Insider Threats in Computer Activities, in Proceedings of the Workshop on Research for Insider Threat, IEEE CS Security and Privacy Workshops, San Francisco, CA, 23-24 May 2013

[Young 2014] Young, W. T., Memory, A., Goldberg, H. G., & Ted, E. (2014, May). Detecting unknown insider threat scenarios. In *Security and Privacy Workshops (SPW), 2014 IEEE* (pp. 277-288). IEEE.

Appendix A – PRODIGAL Installation Guide

PRODIGAL comprises back-end anomaly detection analytics that evaluate computer usage data collected by SureView and produce single scores for each user on each day that indicate the likelihood of potential malicious insider threat activities. PRODIGAL analytics consist of dozens of detectors that look for specific patterns of behavior (e.g., copies of files to removable media) and more general anomalies in the data. PRODIGAL enables analysts to focus on the data most likely to contain true anomalies indicative of insider threats.

1.1. System Requirements.

The document presents the hardware and software requirements and describes how features are specified (i.e., the data requirements) in PRODIGAL.

Hardware and software requirements: The following **Table A-1** summarizes the hardware and software requirements for installing PRODIGAL.

Table 1-1. *PRODIGAL hardware and software architecture requirements.*

Hardware	**Software (*provided by leidos)**
Red Hat 64 OS	Java 1.7
16 cores	JMS *
128 GB RAM	MySQL
8TB+ hard drive space (solid state)	Apache Tomcat 8
	R *
	Anaconda Python*
	Lisp *

The guidelines for installing the PRODIGAL system (i.e., the extract, transform, and load [ETL] workflow and analytics) follow at the end of this document.

Feature specification/data: Features in PRODIGAL are derived from data collected in SureView. PRODIGAL analytics support features specified from count-based observations (e.g., the number of logon events by user, the number of email sent by a given user). Feature design incorporated domain knowledge provided by subject matter experts and are intended to reflect behaviors of relevance to known and unknown but plausible insider threat scenarios. Features consist of an entity extent (e.g., user, file, workstation ID) and a temporal extent. The default temporal extent in PRODIGAL is a user day, which is defined as a 24-hour period (i.e., not a business day).

Currently, there are 111 features in PRODIGAL based on email, file, groups (based on LDAP information and shared entities such as printers, files, and workstations), logon, printer, process, and URL data types. PRODIGAL also has over 20 ratio features that compare activities within and across data types (e.g., the ratio of the number of all URL events to the number of URL upload events). **Table A-2** summarizes table below summarizes the features already specified in PRODIGAL.

Table A-2. *PRODIGAL feature specifications.*

SV Observation Type	Number of Features	Examples
Email	18	Count of attachments on sent emails
File	28	Count of file events to removable drives
Group	11	Shared printers
Login	4	Count of distinct workstations logged onto
Printer	9	Count of print jobs submitted
URL	13	Count of Blacklist events
Ratio	28	Ratio of file events on removable drives to all file events Ratio of URL uploads to URL downloads Ratio of distinct removable drives to URL upload/download events

Operators may customize features (by either modifying existing features or specifying new ones) in PRODIGAL. At this point, feature customization requires technical assistance from the PRODIGAL engineering team. Existing features may be modified based on available SureView parameters (e.g., adding/removing observation types or previously unused attributes). The primary requirement for defining new features in PRODIGAL is that they must be expressed as a count (e.g., the number of times an event occurs). Non-SureView data types can be integrated into the PRODIGAL ETL workflow but the attributes of the non-SureView data types have to relate to those in the SureView data. For example, facility access records (e.g., badge-in/badge-out data) can be integrated with PRODIGAL's ETL workflow so long as entities in the facility access records are also present in the SureView data.

1.2. Installation Guidelines.

NOTE: *These Install Instructions assume that Centos 6 and Java 7 have been installed on your system (Centos Version 6.5 has been tested) and the user "leidos" has been created with group id of "leidos".*[1]

MySQL Installation and Configuration

1. Remove the current version of mysql from the server.
    a. sudo rpm –e –nodeps mysql-libs
2. Install the latest mysql server rpm.
    a. sudo rpm –i mysql-server-5.6.20-1.linux_glibc2.5.x86_64.rpm
3. Install the latest mysql client rpm
    a. sudo rpm –i mysql-client-5.6.20-1.linux_glibc2.5.x86_64.rpm
4. Start the mysql server.
    a. sudo service mysql start
5. Change the root password for mysql
    a. There will be a temporary password in the /root/.mysql_secret file
    b. Run the secure installation script which will allow you to change the root password:
        i. /usr/bin/mysql_secure_installation

[1] Java may be obtained from http://www.oracle.com/technetwork/java/javase/downloads/java-se-jre-7-download-432155.html. The Linux x64 - RPM Installer should be used.

   ii. Enter the temporary root password from above when prompted
   iii. Create a new root password: **Y**
   iv. Enter the new mysql root password
   v. Confirm the new mysql root password by reentering it.
   vi. Remove anonymous users: **Y**
   vii. Disallow root login remotely: **Y**
   viii. Remove Test DataBase and access to it: **Y**
   ix. Reload privilege tables: **Y**
6. Create the prodgialobservation and prodigalresult databases by performing the following:
   a. mysql –u root –p –e 'create database prodigalobservation'
      i. Enter mysql root password when prompted
   b. mysql –u root –p –e 'create database prodigalresult'
      i. Enter the mysql root password when prompted
7. Create a MySQL (non-root) user to give access to these database tables.
   a. mysql –u root –p –e "create user '***someuser***'@'%' IDENTIFIED BY '***somepassword***'"
      i. where ***someuser*** is the name of the desired db user and ***somepassword*** is the corresponding password for that user
8. Grant the appropriate permissions to the newly created DB User
   a. mysql –u root –p –e "GRANT ALL PRIVILEGES ON *.* TO '***someuser***'@'%'"
      i. where ***someuser*** is the name of the user created in the previous step.
9. Install the mysqlConfiguration RPM
   a. sudo rpm –i mysqlConfiguration-1.0.1.x86_64.rpm
10. Run the installMysql script to create the database tables and partitions.
   a. /teams/saic/prodigal/dbScripts/installMysql.sql
   b. Enter the mysql user created above and the respective password when prompted.
      i. **Note:** Password will be requested 4 times

Active MQ Installation and Configuration

1. Install the wrapper and activeMQ rpms
   a. sudo rpm –i jmsConfiguration-1.0-1.x86_64.rpm
2. Start ActiveMQ
   a. cd /opt/activemq/apache-activemq-5.9.0/bin
   b. ./activemq start

Tomcat Installation and Configuration

1. Install the tomcat installation and the required security certificates
   a. sudo rpm –i tomcatConfiguration-1.0-1.x86_64.rpm
   b. cd /etc/pki/java
   c. keytool -import -alias ca -file /etc/ssl/tomcat.cer -keystore cacerts –storepass changeit
2. Start the tomcat application
   a. sudo service tomcat start

Framework Installation

1. Install the framework RPM
   a. sudo rpm –i framework-1.9-1.x86_64.rpm

2. Copy the tomcat war files into the tomcat webapps directory
    a. cp /teams/saic/prodigal/bin/*.war /opt/tomcat/apache-tomcat-7.0.55/webapps
        i. May need to use sudo depending on tomcat permissions

Algorithm Installation

1. Install the GTRI RPM
    a. sudo rpm -i GTRI-1.0-1.x86_64.rpm
2. Install the OSU RPM
    a. sudo rpm -i OSU-1.0-1.x86_64.rpm
3. Install the UMASS RPM
    a. sudo rpm -i UMASS-1.0-1.x86_64.rpm
4. Install the CMU RPM
    a. sudo rpm -i CMU-1.0-1.x86_64.rpm
5. Install the Leidos RPM
    a. sudo rpm -i Leidos-1.0-1.x86_64.rpm

PRODIGAL Web Configuration

Visit this page in your web browser:

https://localhost:8443/prodigal-security-service/form.jsp

You may need to add a security exception the first time you visit this page. For example, with Firefox:

1. Click “Add Exception”
2. Uncheck “Permanently store this exception”
3. Click “Confirm Security Exception”

*You should see a form like **Figure A-1**.*

Fill in the form as follows. In Prodigal Configuration:

1. Username: prodigal
2. Password: prodigal
3. IP Address:
4. Port Number: 8080
5. Observation Database: prodigalobservation
6. Results Database: prodigalresult

In Sureview Configuration:

1. Username:
2. Password:
3. IP Address:
4. Port Number:
5. SID:

In JMS Configuration:

1. IP Address:
2. Port Number:

Then click “Submit Form.”

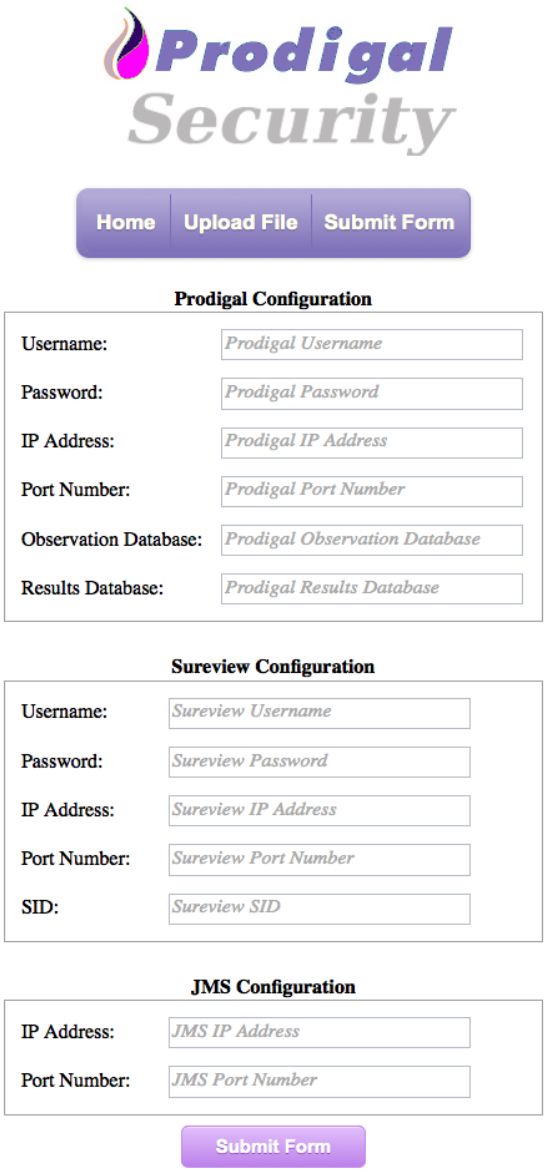


Figure A-1. *PRODIGAL Web Configuration template.*

Appendix B – PRODIGAL Introduction for Transition Partners

1.1. The Insider Threat Problem and PRODIGAL

DARPA's Anomaly Detection At Multiple Scales (ADAMS) program sponsored research in the characterization and detection of anomalies in large datasets. The application domain was insider threat detection; specifically, insider behaviors manifested in computer usage data. Malicious insider activities in the context of the ADAMS program were characterized by low signal-to-noise ratios between malicious and non-malicious activities; patterns of activity that evolve over weeks; and collaboration between an unknown number of groups that consisted of users who spanned organizational, functional, and project boundaries.

The ADAMS program began in 2011 and ended in 2016, and was managed out of DARPA's Information Innovation Office (I2O). Leidos' solution developed for the ADAMS problem is PRODIGAL (PROactive Detection of Insider Threats with Graph Analysis and Learning). Members of the PRODIGAL team included researchers from Leidos, Oregon State University, the University of Massachusetts at Amherst, Georgia Tech University, and Carnegie Mellon University.

PRODIGAL aggregates and compares observations over users, peer groups, and time periods, yielding features that are relevant to detecting malicious insiders based on the semantics (i.e., metadata) and structure of the data. PRODIGAL features include transactions (e.g., logons), interactions (e.g., emails, IMs), person-person and person-resource (e.g., printer usage) graphs, and counts and ratios (e.g., URL uploads to downloads). PRODIGAL applies multiple (>100) detectors including scenario-inspired detectors for known scenarios and machine learning-based anomaly detectors for unknown scenarios.

PRODIGAL combines the results from the multiple detectors using an ensemble technique that provides the analyst with a single score for each user, for each day, which serves as a starting point for further investigation. A practical goal of the PRODIGAL approach was to find the most unusual users, when those users are unusual, and provide explanations that are intuitive to analysts as to the activities that makes users so unusual. By providing the analyst with a list of the most anomalous users, PRODIGAL reduces the search space and focused the analyst on the data that are most likely to contain true anomalies indicative of insider threat behaviors.

1.2. Overview of the PRODIGAL Prototype System

The PRODIGAL prototype system consists of three major components: extract, transform, and load (ETL); analytics; and the analyst interface (AI). The ETL component ingests raw observations collected from user activity monitoring technology and extracts features for processing by the analytics. In the context of PRODIGAL, features combine attributes of UAM data types (e.g., the count of URL uploads, the count of file copies to removable media) that expose malicious insider activities. We derive features by "user day" – that is, we aggregate activities by data type, user, and calendar day. PRODIGAL specifies over 100 features that, given the incorporation of domain knowledge in the feature design process, may be customized given activities of interest.

PRODIGAL analytics encompass a suite of diverse, machine learning-based and domain-inspired detectors and the ensemble algorithm. In PRODIGAL, a detector connotes a specific algorithm that ingests features given a comparison group over a period of time. Machine learning-based detectors include graph analytics, vector space models, decision trees, Gaussian

Mixture Models, and density space estimation. PRODIGAL includes machine learning techniques adapted to look for specific known or suspected complex insider threat behaviors. Examples of domain-inspired detectors include the Intellectual Property (IP) Thief, Saboteur, the Careless User, and the Rager. The PRODIGAL ensemble algorithm, also included in the analytics suite, represents an extension to the state-of-the-art technology in unsupervised anomaly detection. The ensemble algorithm, a machine-learning approach, uses the output of the 100+ detectors as input and produces a single score based on the detectors whose output consensus correlates best among themselves and to a surrogate answer key.

The Analyst Interface (AI), a lightweight, web-based graphic visualization application, presents the results of the analytics suite without overwhelming the analyst with details of the underlying detectors. The AI enables the user to drill-down into an ensemble score and view the results of the individual detectors and features in domain terms, allowing the analyst to understanding intuitively which users are the most unusual compared to their peers, when, and the specific actions that made the users so anomalous. The AI also allows analysts to focus on users' behavior over multiple days, supporting inferences into complex behaviors. **Figure B-1** depicts process by which PRODIGAL's components generate anomalies from user activity monitoring data.

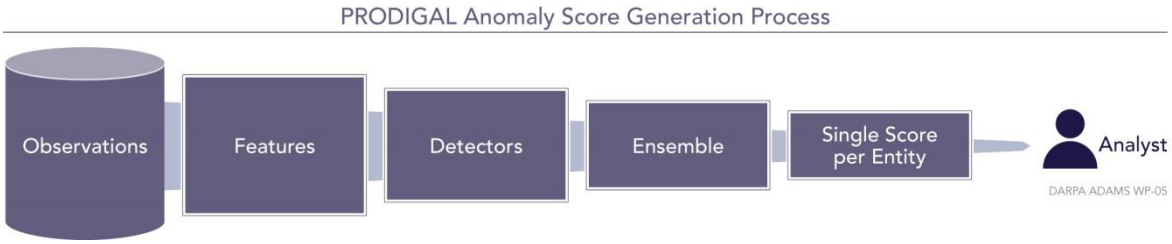


Figure B-1. *PRODIGAL encompasses an end-to-end workflow*

1.3. Background: ADAMS program evaluations

The ADAMS program evaluation approach represents a significant contribution in the area of insider threat detection research. The ADAMS program featured a closed data laboratory as a test bed and UAM data collected through a single instance of SureView from a 5500-person organization. The data consisted of all actions that occurred on users' workstations for 24 months. Estimating that users have, on average, 1000 observations a day, each month contained 165 billion observations. The data provider anonymized and hashed all users to protect personally-identifiable information (PII), and all processing and analysis occurred in the data laboratory.

Separately from the data collection process, the evaluation team (Carnegie Mellon University's Computer Emergency Response Team/Software Engineering Institute) developed scenarios reflecting their field research in insider threat. Scenarios encapsulated specific insider threat behaviors comprised of complex, multi-stage activities. The Red Team developed and superimposed observables on to the actions of real users identified as appropriate to particular roles in the scenario, blending the synthetic observations with the users' normal ones.

The evaluation team designed the scenarios to reflect their research, which indicate that insider threat behaviors constitute a very small portion of malicious insiders' observable behaviors, the Red Team inserted a relatively small number of malicious activities compared to the users'

natural observation. For example, in two scenarios the number of inserted malicious behaviors (~500) represented less than 1 percent of the user's total behaviors for a month. Likewise, the number of insiders identified as being malicious by the Red Team was also less than 1 percent of the total population (e.g., 12 malicious insiders out of the population of 5500) in each data set. Examples of malicious insider behavior addressed in the scenario include theft of intellectual property by technical personnel; insertion of malware by privileged users; unauthorized, surreptitious removal of sensitive files by privileged users using removable media; spoofing of privileged users' credentials to commit fraud; and transfer of sensitive files using email and uploads to file sharing websites. The Red Team, to avoid evaluation bias, designed its scenarios independently of the detection methods developed by the research teams. Likewise, PRODIGAL researchers did not review scenario specifics nor train our detectors on the test data to avoid over-fitting. **Figure B-2** provides examples of ADAMS scenarios.

Name (#/instances)	Synopsis	Name (#/instances)	Synopsis
Anomalous Encryption (2)	An insider passes proprietary information to an outsider, secretly encrypting files and emailing them from his work email to his personal email.	Indecent RFP (1)	An insider uses an inappropriate relationship with another insider to influence illegally vendor selection for a contract for personal financial gain.
Bollywood Breakdown (1)	An insider is convinced by a foreign official posing as a hospital administrator to steal sensitive files in exchange for preferential medical treatment for a relative abroad.	Insider Startup (6)	Three co-conspirators collude to steal company IP. They coordinate the synchronized theft of proprietary information before leaving the company.
Breaking the Stovepipe (3)	An insider accidentally exposes his access to proprietary information of company A to company B. Company B's contact bribes the insider to share the information.	Layoff Logic Bomb (2)	An insider, concerned about worried layoffs, uploads a logic bomb into the IT system that will "detonate" unless he disables it.
Bona Fides (2)	Espionage volunteer prints a bona fides package and takes it to a foreign embassy.	Manning Up (1)	An insider emulates Bradley Manning and researching similar techniques while at work .
Byte Me (2)	An insider uses his role privileges in the badge entry system and sells special access badges to smokers for an unauthorized doorway that is closer to the smoking area. The insider regularly deletes database records related to these events.	Manning Up Redux (1)	An insider emulates Bradley Manning and researches detection counter-measures and scripts code that will upload large amounts of files through a custom DNS
Byte Me Middleman (1)	A variation of Byte Me where all illicit electronic communication with the insider goes through a co-conspirator, obscuring the role of the insider in the activity.	Masquerading 2 (2)	Subject sets up a rogue SSH server on another user's machine. They also make a copy of the local Windows password file and copy the file off over the network.
Circumventing SureView (2)	A user circumvents SureView monitoring to commit a crime.	Naughty by Proxy (2)	A disgruntled insider seeks revenge and logs on to their manager's computer and visits questionable websites.
Credit Czech (1)	An insider runs an illicit business trafficking stolen credit card numbers using the organization's IT resources. He acts as a middleman between various external buyers and a Russian operative who buys the collected numbers.	Outsourcer's Apprentice (2)	An insider in a software development role outsources tasks (and provides his credentials and remote access to the network) to an outsider.
Czech Mate (1)	Similar to Credit Czech, but the new protocol calls for twice-daily emails insider's Russian counterpart in order to keep the operation alive.	Passed Over (2)	An insider learns that their project is being phased out in a re-org, becomes extremely disgruntled, demands and threatens their managers, and installs malware on several machines before submitting a resignation notice.
Exfiltration of Sensitive Data using Screenshots (3)	An insider steals proprietary/sensitive docs by taking screenshots of specific pages, recursively encrypting the files, and emailing them to a webmail address.	Selling Login Credentials (1)	One insider who has system privileges sells (with the unwitting help of two other insiders) dummy accounts and temporary access to outsiders.
Exfiltration prior to Termination	An insider is leaving the company and decides to take all of their emails and files with them.	Snowed In	A system administrator is reprimanded for reporting a website security vulnerability. He then discovers sensitive program and exfiltrates the document while performing with legitimate duties.
From Belarus with Love (1)	An insider becomes romantically involved with an outsider at a conference. The outsider uses photographs of the encounters to blackmail subject into delivering proprietary information.	Survivor's Burden (3)	An insider, after experiencing a logic bomb and being passed over for promotion, steals IP that he then uploads to DropBox.
Gift Card Bonanza (1)	Under financial pressure due to a relative's medical expenses, an insider uses a co-worker's email addresses to enter paying surveys and enlists help of email administrator to hide the misuse of IT resources.	What's the Big Deal (1)	An insider uses a cloud storage site for convenience but later learns that the site has been hacked. The insider does not report the activity and does not delete sensitive files from the site.
Hiding Undue Affluence (1)	An employee possesses undue affluence because of ongoing espionage activity. They need to hide the existence of the money from investigators and they perform research on how to do so.		

Figure 48. *Malicious insider scenarios developed to evaluate ADAMS research*

1.3.1. Evaluation approach and metrics

The Red Team created 28 distinct scenarios, varying elements of the scenarios to produce variants (e.g., three insiders collaborate to steal sensitive files using personal webmail, then five insiders steal sensitive files using a combination of removable media and printing before or after core business hours). The Red Team produced over 85 data sets in total (i.e., all scenario variants), inserting multiple data sets covering different scenarios in each month of test data from September 2012 to October 2014. Instances of Red Team scenarios are limited to one month duration and inserted as targets each calendar month. (CERT has found that 2/3 of known insider threat scenarios evolve over less than one month.) This allows for consistent, independent experiments.

For each data set, the Red Team provided an answer key, a list of the observations that the Red Team inserted as the "malicious" activities that specific users undertake which served as ground truth when assessing system output. Researchers received credit for identifying users' malicious activities (i.e., the Red Team inserts) but not users' other, "non-malicious" days. Thus only the behaviors directly associated with the insider threat activities encompassed by the scenario were "hits" when examining system output and the users' other activities (i.e., non-synthetic observations) were false positives.

Metrics were chosen to measure both detection accuracy of the individual algorithms and their contribution to the overall task of providing leads to an analyst. Traditional measures of utility (e.g., precision, recall, and the F1 measure) did not provide meaningful insights into system performance for the insider threat problem. However, given the small number of both malicious insiders and their nefarious activities, we concluded that measures of performance were not insightful. The false positive rate is exceptionally high given the imbalance issue between the two classes of the dependent variable (i.e., 1 = malicious insider user day, 0 = non-malicious user day). Key metrics for PRODIGAL are compute the curve and area under the Receiver Operator Characteristic (ROC) and lift.

ROC curve analysis, which measures the correctly classified instances in the context of the false alarm, was developed during World War II to assess the performance of radar operators at distinguishing aircraft from other signals (e.g., flocks of birds). ROC curve analysis is commonly applied in machine learning for assessing the utility of classifiers, yet the interpretability of the technique still bears relevance for operational settings. Area under the curve (AUC) is the geometric area (a value from zero to 1) of the space below the plot, and characterizes the utility of a classifier at identifying a target compared to a random choice. Thus AUC is insightful for the insider threat detection domain as the metric characterizes the performance of a classifier in a manner consistent with that of a human analyst in real-world settings; correct detection of threats earlier is preferable.

Another metric used was lift to analyze PRODIGAL's performance. In data mining, lift characterizes the ability of a classifier to enhance the likelihood of instances of the target class given a subset of the data compared to random selection across the entire data set. Lift is often used in marketing. For example, firms use focused tactics (e.g., emailing coupons to shoppers) to increase sales volume among the target group (e.g., emailing coupons or advertising discounted items to shoppers who purchased particular items recently). In the context of PRODIGAL, lift characterized the ability of the system to find malicious insiders in subsets of the population – that is, the increase in predictive value that a detector showed for finding

malicious insiders in the top 50 anomalous user days compared to of the likelihood of detecting the target class in a normal distribution. Lift is measured as a continuous variable, with the value representing the number of times the detector is at finding malicious insiders than random choice.

1.3.2. Evaluation Results

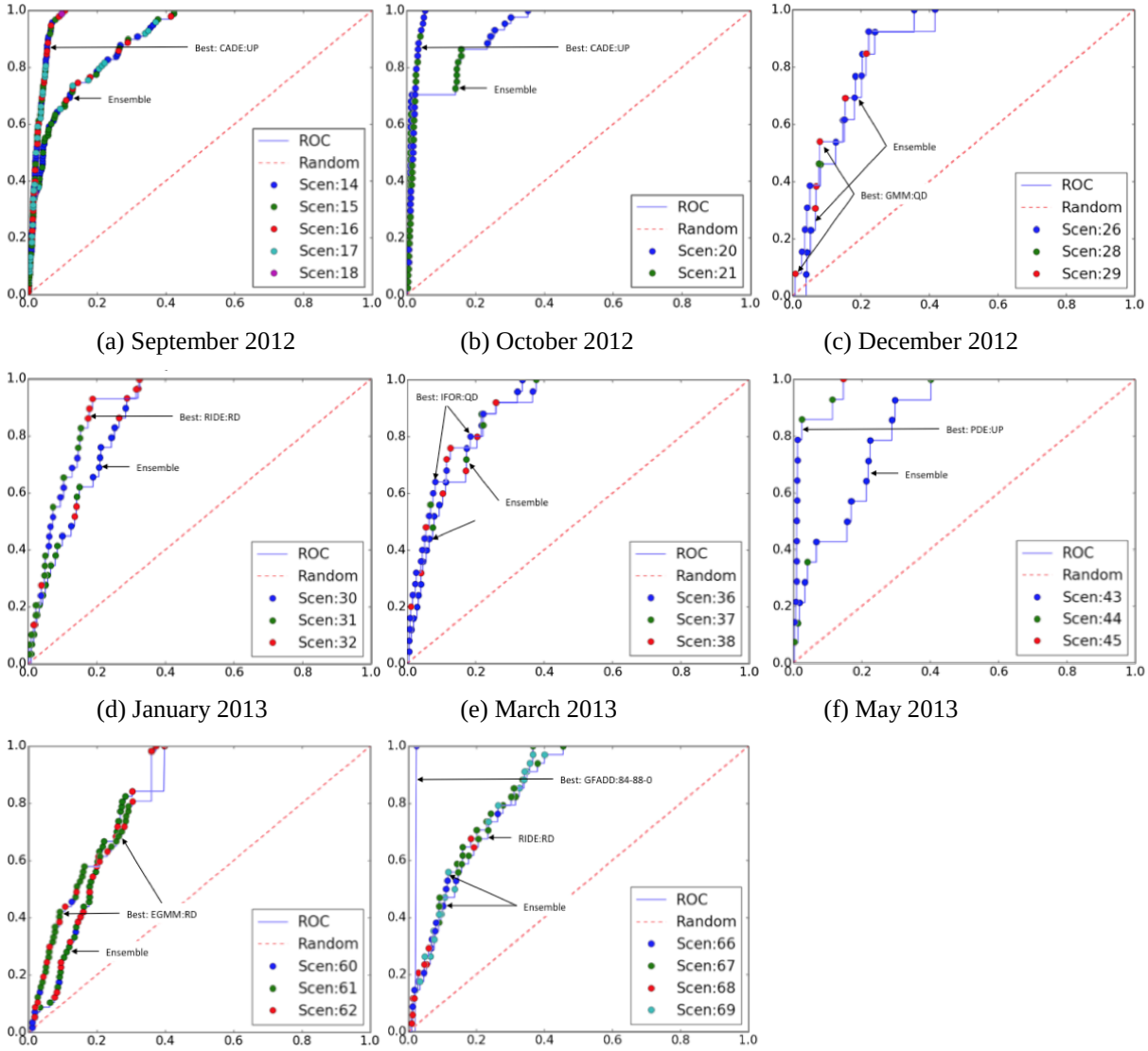
Analysis of system performance addressed the ability of PRODIGAL to detect the days in which Red Team users conducted malicious activities with no prior knowledge of the scenario details. Given our focus, we reported the area under the ROC curve and lift for each data set for the ensemble technique, the single best detector, and the ratio of the ensemble's performance to that of the best detector. As the scenarios details varied from month to month, and our approach was unsupervised machine learning-based, the best detector varied from month to month as well, the ratio of the ensemble's performance to that of the best detector gave insight into the reliability of our ensemble technique. The AUC of the ensemble technique was within 5% of the AUC of the best detector. **Figure B-3** presents PRODIGAL's results over multiple months of test data and various scenarios.

Month	*Ensemble AUC*	*Best Detector*	*Best Det. AUC*	*Ens. / Best RT*	*Scenarios*
12-Sep	0.8973	CADE:UP	0.9703	92.47%	Circumventing SureView Insider Startup
12-Oct	0.9319	CADE:UP	0.9804	95.05%	Insider Startup
12-Nov	0.7542	File	0.7895	95.53%	Anomalous Encryption, Layoff Logic Bomb, Masquerading 2
12-Dec	0.8646	GMM:QD	0.8677	99.64%	Anomalous Encryption, Layoff Logic Bomb, Outsourcer's Apprentice
13-Jan	0.8594	RIDE:RD	0.9015	95.34%	Hiding Undue Affluence, Outsourcer's Apprentice, Survivor's Burden
13-Feb	0.7632	EGMM:QD	0.7793	97.94%	Bona Fides, Manning Up, Survivor's Burden
13-Mar	0.8853	IFOR:QD	0.8963	98.77%	Bona Fides, Hiding Undue Affluence, Manning Up Redux
13-Apr	0.8635	RIDE:QD	0.8619	100.19%	Circ. SureView, Indecent RFP, Selling Login Cred., Survivor's Burden
13-May	0.8469	PDE:UP	0.9718	87.14%	Credit Czech, Exfiltration Prior to Termination
13-Jun	0.8852	IFOR:QD	0.9103	97.24%	Czech Mate, Exfiltration of Sensitive Data Using Screenshots
13-Jul	0.8498	RIDE:RD	0.8769	96.90%	Breaking the Stovepipe, Snowed In
13-Oct	0.8938	GMM:RD	0.8972	99.62%	Breaking the Stovepipe, Snowed In

Month	*Ensemble AUC*	*Best Detector*	*Best Det. AUC*	*Ens. / Best RT*	*Scenarios*
13-Nov	0.8479	RIDE:RD	0.8459	100.23%	Byte Me, Naughty by Proxy
13-Dec	0.8034	EGMM:RD	0.828	97.02%	Byte Me Middleman, Indecent RFP 2, Passed Over
14-Jan	0.8425	IFOR:RD	0.8242	102.22%	From Belarus With Love, Passed Over, What's the Big Deal
14-Feb	0.847	GFADD:84-88-0	0.9775	86.65%	Bollywood Breakdown, Breaking the Stovepipe, Gift Card Bonanza, Naughty by Proxy

Figure 49. *Ensemble results and best individual detector result by month and scenario*

Figure B-4 is a set of ROC curve plots depict the results presented in Figure 3 and illustrate our analysis approach. Each plot represents a month of data into which the Red Team inserted between two and five scenarios. The plot lists scenarios by the designation given to them by the Red Team (reference **Figure B-2**).



(a) September 2012 (b) October 2012 (c) December 2012

(d) January 2013 (e) March 2013 (f) May 2013

(g) December 2013 (e) February 2014

Figure B-4. *ROC curves vs. all RT inserts for the Ensemble and the best detector for various months*

1.4. Findings and future research

Analysis of PRODIGAL's performance on the test data indicate that our approach, which consists of a diverse suite of unsupervised, machine-learning based detectors, show promise given the presence of unknown malicious insider threat behaviors in user activity monitoring data at scale (i.e., extremely low signal-to-noise ratios of threat to normal activity and a very small number of malicious users compared to the entire population). Our ensemble approach appears robust, scoring within 5% (confidence level of +/- 1.5%) of the best detector for any scenario, where the scenario details vary from period to period and are unknown prior to processing. PRODIGAL's performance against the test data demonstrated that our analytics suite enriches the target environment and narrows the insider threat search space to identify the

most unusual users, when they are unusual, and highlights their most unusual activities. The features that PRODIGAL extracts from user activity monitoring data Incorporate domain knowledge of previously observed and suspected threat behaviors to find similar or analogous patterns that enables the detection of unusual behaviors based on combinations of activities that, individually, would not be noticed by tripwire-based approaches. The innovative ensemble algorithm to combine results from the analytics suite into a single score, providing analysts with a starting point for investigation.

While PRODIGAL showed potential utility in research settings, future work lies in evaluations that demonstrate the system's capabilities in operational settings. This focus enables potential transition partners the opportunity to assess the suitability and utility of PRODIGAL as the back-end analytics system in the context of insider threat detection and investigation in the context of an IC environment. Assessments of PRODIGAL using real, Intelligence Community (IC) data and operated by real analysts enable the development of metrics for characterizing the utility of PRODIGAL for insider threat detection and investigation activities. Evaluation of PRODIGAL's capabilities in the context of operational workflows also facilitates exploration of concepts of operations for integrating analytics with user activity monitoring data, as well as opportunities for incorporating new data types (e.g., badge-in/badge-out data). Operational evaluation also gives insight into the level of effort for installing, configuring, and operating PRODIGAL extract, transform, and load (ETL) component, as well as integration requirements for the implementation of PRODIGAL in an architecture.