**Ideas:**

* Анализ состояний системы на основе различных уровней мониторинга (pcap + logs analysis).

**-** Формулировка метрик нефункционального тестирования с точки зрения задачи машинного обучения.

**Problem:**

Хочется получить информацию о состоянии системы за определенный период времени. Для этого анализируются логи системы. По этому поводу написано большое количество работ: как распарсить[1][2][3], структурировать[4], извлечь features[5], построить модель которая будет выявлять или прогнозировать аномалии в работе системы. Модель может быть как статистическая (Principal component analysis) так и с использованием машинного обучения (supervised[6][7] или unsupervised[8][9][10][11]). В этой области прмиеняется широкий диапазон различных моделей: Support Vector Machine, Seq2Seq, LSTM и различные модификации с попыткой улучшения качества прогноза и обнаружения аномалий. Далее следует оценить построенную модель, на сколько точно она выявляет аномалии, ее способность в обобщению, масштабированию. И в конечном счете визуализировать и интерпретировать полученный с ее помощью результат[12] (Иногда недостаточно получить точный прогноз, хочется проследить каким образом модель приняла конкретное решение, это особенно важно для некоторых областей: медицина, авиапромышленность, космос).  
 Такой подход может хорошо сработать для систем которые прошли тщательное тестирование и можно предположить что система логирования работает корректно (опять же в ней могут быть ошибки). Поэтому единственный (не очень надежный) источник данных для постоения систем поиска аномалий (классический или с использованием машинного обучения) не является надежным (особенно для критически важный систем). И уж тем более такой подход не приемлем для систем которые находятся на этапе тестирования.  
 Также существует подход основанный на парсинге source code системы для извлечения операторов логирования и на их основе структирировать логи [][]. Такой способ может быть эффективнее чем оперирование только с логами системы, но таит в себе две проблемы: 1) В случае независимого тестирования сторонняя компания не имеет доступа к исходному коду (и вообще редко команды тестирования видят исходные коды) 2) В самом исходном коде уже могут содержаться ошибки.

**Solution:**

Использовать для анализа несколько независимых источников данных. Например анализ логов + анализ pcap файлов (+ какие-то метрики системы, память, процессор). Применение такого комплексного подхода позволяет избежать использование единственного источника информации о состоянии системы. А так же найти дефекты в механизмах мониторинга и логирования самой системы.

**Our contribution:**

Our contribution is a general methodology that allows machine learning and information retrieval techniques to be applied for analysis the state of the under testing system using several undependent data sources.

**References**

[1] Min Du and Feifei Li. 2016. Spell: Streaming Parsing of System Event Logs. In Proc. IEEE International Conference on Data Mining (ICDM). 859–864.  
[2] A. Makanju, A. N. Zincir-Heywood, and E. E. Milios, “A lightweight algorithm for message type extraction in system application logs,” TKDE, 2012.  
[3] Y. Li, Y. Wang, Z. Zhang, Y. Wang, D. Ma, and J. Huang., “A novel fast and memory efficient parallel mlcs algorithm for longer and large-scale sequences alignments,” in ICDE, 2016.  
[4] Pinjia He, Jieming Zhu, Shilin He, Jian Li, and Michael R Lyu. 2016. An evaluation study on log parsing and its use in log mining. In Proc. International Conference on Dependable Systems and Networks (DSN). 654–661.  
[5] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. 2009. Detecting large-scale system problems by mining console logs. In Proc. ACM Symposium on Operating Systems Principles (SOSP). 117–132  
[6] Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In Proc. Neural Information Processing Systems Conference (NIPS). 3079–3087.  
[7] Gavai, G.; Sricharan, K.; Gunning, D.; Hanley, J.; Singhal, M.; and Rolleston, R. 2015. Supervised and unsupervised methods to detect insider threat from enterprise social and online activity data. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications 6(4):47–63.  
[8] J.-G. Lou, Q. Fu, S. Yang, Y. Xu, and J. Li, “Mining invariants from console logs for system problem detection.” in USENIX ATC, 2010.  
[9] J.-G. Lou, Q. Fu, S. Yang, J. Li, and B. Wu, “Mining program workflow from interleaved traces,” in SIGKDD, 2010.  
[10] Shilin He, Jieming Zhu, Pinjia He, and Michael R Lyu. 2016. Experience Report: System Log Analysis for Anomaly Detection. In Proc. In  
[11] Tuor, A.; Kaplan, S.; Hutchinson, B.; Nichols, N.; and Robinson, S. 2017. Deep learning for unsupervised insider threat detection in structured cybersecurity data streams. In Artificial Intelligence for Cybersecurity Workshop at AAAI.  
[12] Andy Brown, Aaron Tuor, Brian Hutchinson, and Nicole Nichols. 2018. Recurrent Neural Network Attention Mechanisms for Interpretable System Log Anomaly Detection. In Proceedings of ACM HPDC (First Workshop On Machine Learning for Computer Systems). ACM, New York, NY, USA, 8 pages.

**Abstract**