CSCE 689 Winter 2016: Project Proposal

Paul Jordan and Chip Van Patten

February 18, 2016

1 Proposal Abstract

Today, network service outages can often result in mission failure. In recent news, network service outages have been the cause of the nationwide grounding of commercial airlines. In the Air Force, this kind of outage would be catastrophic. We depend upon the same technology as our civilian counterparts, but our mission is National Security.

Fortunately, an immense amount of work has gone into developing machine learning algorithms that can warn us when failure is imminent ([SLM10, IVD15, IV14, Wat14]) and in many cases take action to avoid the failure. Unfortunately, much of this work has gone unused and further, does not work well in distributed environments. In 2015, a framework was proposed in [IVD15] to make implementing failure prediction algorithms in production environments easier. Sadly, the framework did not address distributed systems. Some work has been done in [Wat14, WOS+12, SWM12] to address the issue of these techniques not working in distributed systems, but these techniques require failure to occur at least once to be effective. For the Air Force, one failure could have unacceptable consequences.

A significant part of the framework defined in [IVD15] is dependent upon being able to generate realistic load against the service under test. Unfortunately, while much work has been done in the way of traffic generation ([AGE+04, ADPF+08, MMS13, VV09, AOPK09, ZPM13]), none of it appears to generate the traffic we would need to load a domain controller sufficiently. Specifically, we need full-stack authentication traffic and many of the generators out there only provide one way communication.

In our work, we intend to enable the extension of the work in [IVD15] by developing a tool that can generate realistic load in a distributed environment for the purpose of training machine learning algorithms to detect system failures in a Microsoft Windows domain. If we are able to predict failure with enough lead-time to enable automated failure avoidance, we can ensure the mission critical systems stay online to enable future mission success.

2 Research Activities

2.1 Groundwork Tasks

We'll need to first develop the tool, setup a virtual test environment, and setup a data collection point. The virtual environment will need to have all of the infrastructure one might find in a domain (Domain Controller, DNS, Mail, client machines, etc...).

2.2 Research development tasks

The tool we plan to write is in Powershell. The tool must be capable of running from a remote controller, so the controller will have to be developed as well.

2.3 Evaluation methods

We will conduct an experiment with the tool to determine how realistic the traffic generated by the tool will be. To do this, we'll have to manually use the client machines to conduct work tasks and collect traffic generated. Then we can compare the manually generated traffic with the traffic generated by our tool. We will also compare the log entries created by manual versus automated use.

2.4 Equipment required

We will need at least one hypervisor to setup our virtual environment (to which we already have access), and licenses for Microsoft Windows Server and Microsoft Windows 7 (that we have already obtained).

2.5 Team required

We (1st Lt Paul Jordan, and 1st Lt Chip Van Patten) plan to conduct this research.

3 Annotated Bibliography

References

[ADPF+08] G. Antichi, A. Di Pietro, D. Ficara, S. Giordano, G. Procissi, and F. Vitucci. Design of a high performance traffic generator on network processor. In *Digital System Design Architectures, Methods and Tools, 2008. DSD '08. 11th EUROMICRO Conference on*, pages 438–441, Sept 2008.

It is suggested in this paper that the currently available, PC-based network traffic generators are not able to adequately replicate realistic traffic on high-speed networks. The authors present their solution of implementing the BRUTE traffic generator on an Intel Network Processor. They only present the proposed design and architecture of their traffic generator solution, leaving actual implementation and performance tests to future work. This paper offers inspiration for possible future iterations and eventual improvements of our method.

[AGE+04] S. Avallone, S. Guadagno, D. Emma, A. Pescape, and G. Ventre. D-ITG distributed Internet traffic generator. In *Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings. First International Conference on the*, pages 316–317, Sept 2004.

This paper provides a new solution to traffic generation. The authors implemented a traffic generator that emulates various network protocols and the Inter Departure Time and Packet Size random variables for the traffic. The authors compare the performance of their software with other traffic generators and provide a brief analysis of the performances. This paper offers another comparison for our method.

[AOPK09] C. Albrecht, C. Osterloh, T. Pionteck, and R. Koch. An Application-Oriented Synthetic Network Traffic Generator. In *22nd European Conference on Modelling and Simulation.*, 2009.

This paper proposes that current traffic generators do not do a good job of testing network hardware in a realistic manner. The authors introduce their tool that generates multiple types of network traffic and varies the ammount and timing of the traffic based on models. They provide results from experiments conducted with their tool along with analysis and comparisons to other traffic generators available. This paper provides a comparison for our method.

[DM06] J. Duraes and H. Madeira. Emulation of software faults: A field data study and a practical approach. *Software Engineering, IEEE Transactions on*, 32(11):849–867, Nov 2006.

This paper suggests that fault injection is used, but is not well understood. The paper goes on to thoroughly analyze and formalize a technique for software fault injection. The paper then conducts a field study to demonstrate how well the method works on real software. The key takeaway here is that it reinforces why we are developing a load generator.

[IDMV13] I. Irrera, J. Duraes, H. Madeira, and M. Vieira. Assessing the impact of virtualization on the generation of failure prediction data. *Proceedings - 6th Latin-American Symposium on Dependable Computing, LADC 2013*, pages 92–97, 2013.

The problem this paper suggests is that in order to conduct online failure prediction using machine learning in real-time, one must train a machine learning algorithm with real failure data. Unfortunately, production systems do not fail very often and injecting failures into production systems is unacceptable. The paper suggests that virtualization is a cheap alternative for injecting faults and collecting failure data to train prediction models. The authors compare failure data collected from live systems with failure data collected from virtualized systems. This paper helps us establish that it's okay to virtualize our experiment.

[IDVM10] I. Irrera, J. Duraes, M. Vieira, and H. Madeira. Towards Identifying the Best Variables for Failure Prediction Using Injection of Realistic Software Faults. *2010 IEEE 16th Pacific Rim International Symposium on Dependable Computing*, pages 3–10, 2010.

The problem presented in this paper is that identifying the proper variables to conduct failure prediction is difficult. The paper presents a methodology for identifying the variables that should be used to predict failure by injecting software faults. The authors then conduct two experiments to test their approach. The key takeaway for our work is that this paper helps establish the need for a load generator in failure prediction.

[IPV13] I. Irrera, C. Pereira, and M. Vieira. The Time Dimension in Predicting Failures: A Case Study. *2013 Sixth Latin-American Symposium on Dependable Computing*, pages 86–91, 2013.

This paper suggests that the time-dimension makes predicting failures using a Support Vector Machine difficult and provides a technique for accounting for this difficulty. The authors then conduct a case study to test their method and report findings. The key takeaway for our work is that it reinforces our reason for developing a load generator.

[IV14] I. Irrera and M. Vieira. A Practical Approach for Generating Failure Data for Assessing and Comparing Failure Prediction Algorithms. *2014 IEEE 20th Pacific Rim International Symposium on Dependable Computing*, pages 86–95, 2014.

This paper suggests that machine learning based failure prediction methods must be trained using known failure data, but those data are hard to find. This paper suggests a method for generating those data using software fault injection and simulated load. The key takeaway for our work is that this helps establish the need for a load generator.

[IVD15] I. Irrera, M. Vieira, and J. Duraes. Adaptive Failure Prediction for Computer Systems: A Framework and a Case Study. *2015 IEEE 16th International Symposium on High Assurance Systems Engineering*, pages 142–149, 2015.

The problem this paper attempts to solve is that many failure prediction techniques have been developed and shown to be effective but they have not been implemented on production machines. The authors present a framework and then demonstrate its efficacy by doing a case study. The key takeaway for our work is that our work enables the extension of this framework for a distributed environment.

[MMS13] S. Molnar, P. Megyesi, and G. Szabo. How to validate traffic generators? In *Communications Workshops (ICC), 2013 IEEE International Conference on*, pages 1340–1344, June 2013.

This paper is more of a survey of network traffic generators and does not provide a solution to a specific problem, but rather advocates that the community adopt a standardized validation method for traffic generating tools. This paper provides insight into a number of methods to validate our tool and gives insight into the numerous validation processes.

[SLM10] F. Salfner, M. Lenk, and M. Malek. A survey of online failure prediction methods. *ACM Comput. Surv.*, 42(3):10:1–10:42, March 2010.

There isn't really a problem statement since this is a survey, but in general, machines tend to fail for many reasons. The methods described in this survey are for predicting failure in real-time (as opposed to before-hand statistical analysis). Again, many approaches and solutions are presented in this survey. Our focus is on the machine learning approaches that focus on error messages (or log messages). The key takeaway for our work is that this helps set the stage for our problem.

[SWM12] M. Sonoda, Y. Watanabe, and Y. Matsumoto. Prediction of failure occurrence time based on system log message pattern learning. *Network Operations and Management Symposium (NOMS), 2012 IEEE*, (4):578–581, 2012.

This paper suggests that failure prediction in cloud data-centers is difficult because there is no guarantee on the order in which reported errors are received. The paper reports results from an experiment conducted using the method presented. This paper offers a comparison for our method.

[VSMM12] V. Vasic, M. Suznjevic, M. Mikuc, and M. Matijasevic. Improving distributed traffic generation performance by using imunes network emulator. In *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on*, pages 1–5, Sept 2012.

There is no specific problem stated in this paper except to improve on the authors' previously implemented network traffic generator. The authors do so by implementing the Integrated Multiprotocol Network Emulator/Simulator as the network emulator used. The authors then report on the results of experiments run using the methodology detailed. This paper will serve as a comparison for our method.

[VV09] K. Vishwanath and A. Vahdat. Swing: Realistic and responsive network traffic generation. *IEEE/ACM Transactions on Networking*, 17(3):712–725, 2009.

The problem this paper suggests is that traffic generators should accurately reflect real-world variances, distributions, bandwidth, and burstiness. The authors present their tool, Swing, which generates traffic based on characteristics observed from a given network's behavior. They present the results of their tool and discuss whether their tool fully achieves the goal introduced by the paper. This paper provides us with a method to compare and contrast our tool and more ways to analyze our results.

[Wat14] Y. Watanabe. Online Failure Prediction in Cloud Datacenters. 50(1):66–71, 2014.

This paper suggests that failure prediction in cloud data-centers is a difficult task. It presents a new way of doing failure prediction in cloud data-centers and then reports results and performance metrics from an experiment conducted using the new method. Our work will compare and contrast this method with ours.

[WOS+12] Y. Watanabe, H. Otsuka, M. Sonoda, S. Kikuchi, and Y. Matsumoto. Online failure prediction in cloud datacenters by real-time message pattern learning. *CloudCom 2012 - Proceedings: 2012 4th IEEE International Conference on Cloud Computing Technology and Science*, pages 504–511, 2012.

In this paper, it is suggested that nodes fail in cloud data centers and conventional machine learning techniques do not work well due to the fact that there is no guarantee as to the order in which reported error messages are received. The paper then presents results from a case study conducted using the method presented. The key takeaway here will be to compare and contrast our method with this one.

[ZPM13] P. Zach, M. Pokorny, and A. Motycka. Design of Software Network Traffic Generator. *Recent Advances in Circuits, Systems, Telecommunications and Control*, pages 244–251, 2013.

The authors of this paper propose that network traffic generators fall into specific categories, each with a specific purpose. They attempt to create a hybrid solution which spans multiple categories. The authors conduct a case study using their tool and provide results and comparisons with other popular traffic generation tools. This paper will provide another method to compare and contrast with ours.