Software Engineering for Infrastructure and Configuration (SEConfig) – Workshop Report

Jürgen Cito
MIT
jcito@mit.edu

Mark Santolucito
Yale University
mark.santolucito@yale.edu

DOI: 3385678.3385686

ABSTRACT

http://doi.acm.org/3385678.3385686

Errors in configurations, rather than source code, have become one of the major causes of system failures, resulting in security vulnerabilities, application outages, and incorrect program executions. We report on the structure and results of the first International Workshop on Software Engineering for Infrastructure and Configuration Code. Our aim in organizing this jworkshop was to

1. INTRODUCTION

Configuration errors (also known as misconfigurations) have become one of the major causes of system failures, resulting in security vulnerabilities, application outages, and incorrect program executions. Building tools for configuration file support, management, and verification has been an active direction of research. As support for configuration files grows, so too does the scope of their application, for example in domains such as Infrastructure as Code.

Accordingly, the first International Workshop on Software Engineering for Infrastructure and Configuration Code (SEConfig) was co-located with the 34th International Conference on Automated Software Engineering (ASE 2019) in San Diego, CA, USA. The workshop provided a venue for researchers and practitioners to come together and discuss the open challenges in the domain of configurations, broadly interpreted. Some of the questions we tackled included: what new security risks develop in the Infrastructure as Code paradigm, and how can we mitigate these? As the definition of a configuration languages is more dynamic than traditional programming languages, how can our tools automatically adapt? How can we verify the correctness of configurations when some errors manifest at system initialization time, while others only manifest under particular system environments? What is the best way to discover configuration setting recommendations and present them to a developer? The goal of the workshop was to gain clarity and specificity on the current open challenges in the area and understand how techniques from industry and various academic fields (e.g., Software Engineering, Verification, Programming Languages) can come together to advance the community's solutions.

Topics of interest included:

1. Infrastructure and configuration code management
2. Specification learning and mining for configurations
3. Infrastructure and Configuration testing and verification
4. Infrastructure as Code and configuration repair
5. Analysis of configuration usage patterns
6. Analysis of configuration failure patterns
7. New languages for configuration

2. FORMAT OF WORKSHOP

The workshop began with an opening keynote given by Shane McIntosh, which addressed some of the relevant issues in the management and analysis of CI/CD data. The morning talks were focused on synthesis and analysis of configurations through formalizations of the configuration systems as state machines [6]. In the afternoon, talks focused on challenges of configuration analysis in particular domains [3, 4]. Specifically, talks looked at areas such as machine learning for in-home monitoring of geriatric patients and optimization of configuration for domain-specific modelling languages.

Throughout the day, there was ample time for unstructured discussion during breaks. In addition, the workshop included a *HackerNews Roulette* session, in which the participants were presented with a number of case studies of configuration difficulties encountered and documented on the discussion board HackerNews [2]. As a group, we discussed key insights to be had from these comments, and worked to contextualize each challenge in the broader context of configuration management research.

At the end of the workshop, we asked participants to break into small workgroups to brainstorm new research directions based on the discussions during the day. Participants split into three groups of roughly equal size and posterboarded their ideas. We list below some of the key takeaways that the groups synthesized.

3. DISCUSSION OUTCOMES

Each group took their own approach to organizing their thoughts on future directions of configuration management and analysis research.

3.1 Group 1

The first group chose to frame their brainstorming sessions in the context of research goals for horizons of one, five, and ten years.

The key phrase for year one idea was *Config Option Eviction*. One of the challenges of increasingly large and complex system is an accumulation of unused configuration options. A tool for configuration option eviction would allow users to automatically suggest configuration options that should be removed from the code base. A proposed approach for such a tool was to use a histogram-like data model of activated configuration options over a training set of configurations to obtain a distribution of which options are most frequently used - and more importantly, how this distribution changes with new versions of the systems. Such an approach to cleaning up unused configuration options could have benefits from both a user and developer perspective. As a consequence of shrinking the activated options, deployed systems may be easier to prove secure - as the attack surface is reduced.

The groups also proposed that this approach could help users with feature discovery - acting as a recommendation system of configuration options for other users.

The first group's five year idea was focused on *inferring roles of configuration options*. Building on the histogram of options, the group proposed to further look at correlations in evolution between options. By inferring groups of configuration options that change in tandem, it may be possible to identify options that have related roles. This may help define a space of optimization for configurations.

Finally, the group closed with a ten year idea on *inferring intention of configurations*. The group aimed to make configuration suggestions based on observations of the development workflow.

Some questions posed by the group included: to what extent could we apply such a system to cross-application configurations? Additionally, feedback from other groups mentioned that this approach relies on the wisdom of the crowd. The group was encouraged to consider to what extent systems should trust the crowd, and how a level of skepticism can be worked into the analysis techniques.

3.2 Group 2

The second group proposed a topic looking at *configuration system migrations*. This idea is addressing the cost of transferring between configuration paradigms, specifically looking at deployment configuration languages. For example, the case of a user is a Chef [1] expert who is asked to migrate to Puppet [5]. We would need to first identify the common pitfalls of configuration migrations (which may or may not be similar to migrating code bases between languages), then work on techniques to mitigate these pitfalls.

Along a similar line of thought, this group considered to what extent do configuration parameter optimization techniques generalize across programming paradigms. For example, what adjustments are needed for a configuration optimization technique in a statically typed language to work in a dynamically typed language. Aside from various programming language paradigms, the group also asked how optimization techniques may need to change over the evolution of language versions.

3.3 Group 3

The third group looked at configuration defaults and their impact on the ability of users to understand the system they are configuring. The group began the design of a study which would compare the learning outcomes of configuring a system between two groups. One group of users would be given access to a system without default values, while the treatment group would attempt to configure a system *with* default values available. The key metric to compare success may be the amount of time it takes to configure the system. The group hypothesized that the differences would lie in long-term learning gains of the system, which would require a longitudinal study design.

4. LOOKING AHEAD

In summary, the SEConfig workshop successfully outlined and explored a number of directions for future research on configuration management. During the workshop, we saw that configuration management is moving beyond software tool configurations, and now encompassing domains such as infrastructure, networks, deployment strategies, and home IoT security. Accordingly, our techniques for configuration management will need to co-evolve with these domains. There has been significant progress in the space of configuration management, but many questions remain unanswered - especially relating the evolution of systems over time and the ability of our tools to generalize over different configuration contexts. Future efforts to automate configuration management will need to handle a new scale and complexity of systems. Gaining a deeper understanding of the way users interact with configurations will be an important aspect to developing such tools. Overall, we are seeing infrastructure and configuration code as an increasing interdisciplinary field of study.

Acknowledgments. We would like to thank the participants of the workshop for their time and engagement, as well as the organizing committee of the Automated Software Engineering conference. This work was supported in part by NSF grants CCF-1302327, CCF-1715387, and CCF-1553168, as well as SNF Grant on "Automated Synthesis and Repair of Infrastructure Code".

References

[1] Chef. `https://github.com/chef/chef`, 2020.

[2] Hacker news. `https://news.ycombinator.com/`, 2020.

[3] Maike Basmer and Timo Kehrer. Encoding adaptability of software engineering tools as algorithm configuration problem: A case study. In *34th IEEE/ACM International Conference on Automated Software Engineering Workshops, ASE Workshops 2019, San Diego, CA, USA, November 11-15, 2019*, pages 86–89. IEEE, 2019.

[4] Lucas Gisselaire, Florian Cario, Quentin Guerre-Berthelot, Bastien Zigmann, Lydie du Bousquet, and Masahide Nakamura. Toward evaluation of deployment architecture of ml-based cyber-physical systems. In *34th IEEE/ACM International Conference on Automated Software Engineering Workshops, ASE Workshops 2019, San Diego, CA, USA, November 11-15, 2019*, pages 90–93. IEEE, 2019.

[5] Puppet Labs. Puppet. `https://github.com/puppetlabs/puppet`, 2020.

[6] Tim Nelson, Natasha Danas, Theophilos Giannakopoulos, and Shriram Krishnamurthi. Synthesizing mutable configurations: Setting up systems for success. In *34th IEEE/ACM International Conference on Automated Software Engineering Workshops, ASE Workshops 2019, San Diego, CA, USA, November 11-15, 2019*, pages 81–85. IEEE, 2019.