# Testing of Cloud Services

Vaggelis Atlidakis

Over the last decade, we have witnessed an explosion in cloud services for hosting software applications (Software-as-a-Service), for building distributed services (Platform-as-a-Service), and for providing general computing infrastructure (Infrastructure-as-a-Service). Thousands of new cloud services have been deployed by cloud platform providers, such as Amazon Web Services [2], Google Cloud [7], and Microsoft Azure [9], and support customers who are modernizing their processes by switching from the complexity of owning and maintaining their own, on-premise Information Technology (IT) infrastructure to instead simply access and pay on demand cutting edge technologies. According to a recent study, in 2020, the public cloud services market is expected to reach around 266 billion U.S. dollars in size and by 2022, its market revenue is forecast to exceed 350 billion U.S. dollars [12].

Today, most cloud services are programmatically accessed by third-party applications [13] and other services [21] through Application Programming Interfaces (APIs) that follow the REpresentational State Transfer (REST) software architectural style [18]. REST APIs are implemented on top of the ubiquitous HTTP and HTTPS protocols, and provide requesting systems (clients) with a uniform and predefined set of stateless operations in order to create, monitor, manage, and delete cloud resources. Furthermore, cloud service developers use interface-description languages, such as the OpenAPI Specification (OAS) [10], to describe and document their REST APIs, including what requests the service can handle through its REST API and the format of those requests, what responses may be received, and the respective response format [5, 6, 1].

The fact that the vast majority of cloud services are accessed through REST APIs that are well-documented with interface-specification languages presents a unique opportunity to build systems that automatically test cloud services through their APIs. When is a cloud service reliable and secure to be publicly deployed? What kinds of software errors may be hiding behind REST APIs? And how critical these errors may be? Automatically answering such questions for production-scale, cloud services still remains an open research challenge and is of paramount importance for a multibillion-dollar market where competing cloud providers seek to avoid reliability and security incidents that will attract negative publicity.

Indeed, despite the rapidly evolving cloud ecosystem, systems for automatically testing cloud services through their REST APIs are still in their infancy. The most sophisticated testing systems currently available for REST APIs capture live API traffic, and then fuzz and replay the recorded traffic with the hope of finding software errors. Many of these fuzzing systems were born as extensions of website testing and scanning tools [4, 11, 3]. Since these REST API testing systems are all recent and not yet widely used, it is still largely unclear how effective they are in finding unknown errors, especially those that hide behind complex APIs and require sequences of events in order to be exposed.

My work leverages the structured usage of cloud services through REST APIs and feedback obtained during interaction with cloud services in the form of response status codes and response objects to build systems that test cloud services through their APIs in an *automatic*, *efficient*, and *learning-based way* [15, 16, 14, 20].

Along with my collaborators, I have built systems that are automatic in that they require minimal manual intervention to test target cloud services, efficient in that they find within a reasonable time-frame (e.g., in few hours) previously-unknown software errors that were beyond the reach of past tools, and learning-based in that they learn without predefined rules or heuristics how to test target cloud services. Test automation is achieved by leveraging the common structure present in the ecosystem of cloud services that are accessed through well-documented REST APIs. Efficiency is achieved by careful consideration of the semantics of the RESTful architectural design style, which allows to generate test sequences that consist of multiple API requests and test target services deeply, and by utilizing feedback obtained during interaction with the target services in order to prune large search spaces and avoid redundant test cases. Finally, the systems I build pursue the avenue of learning-based program analysis and leverage recent advancements in the area of machine learning [17, 22] in order to learn from past tests common usage patterns of target cloud services and generate new tests.

My systems have been used to test various production-scale, open-source and proprietary cloud services ,such as GitLab [19], and Azure [9] and Office-365 [8] cloud services, and have lead to tens of bug fixes so far.

## References

[1] Amazon API Gateway. https://aws.amazon.com/about-aws/whats-new/2018/09/amazon-api-gateway-adds-support-for-openapi-3-api-specification/. (Accessed on 06-02-2020).

[2] Amazon Web Services (AWS). https://aws.amazon.com/. (Accessed on 06-02-2020).

[3] APIFuzzer. https://github.com/KissPeter/APIFuzzer. (Accessed on 06-02-2020).

[4] AppSpider. https://www.rapid7.com/products/appspider. (Accessed on 06-02-2020).

[5] Azure REST API Specifications. https://github.com/Azure/azure-rest-api-specs. (Accessed on 06-02-2020).

[6] Google API Discovery Service. https://developers.google.com/discovery/. (Accessed on 06-02-2020).

[7] Google Cloud. https://cloud.google.com/. (Accessed on 06-02-2020).

[8] Microsoft 365. https://www.microsoft.com/microsoft-365. (Accessed on 06-02-2020).

[9] Microsoft Azure. https://azure.microsoft.com/en-us/. (Accessed on 06-02-2020).

[10] OpenAPI Specification. https://swagger.io/specification/. (Accessed on 06-02-2020).

[11] Qualys Web Application Scanning (WAS). https://www.qualys.com/apps/web-app-scanning/. (Accessed on 06-02-2020).

[12] "Size of the public cloud computing services market from 2009 to 2022". https://www.statista.com/statistics/273818/global-revenue-generated-with-cloud-computing-since-2009/. (Accessed on 06-02-2020).

[13] S. Allamaraju. *RESTful Web Services Cookbook*. O'Reilly, 2010.

[14] V. Atlidakis, R. Geambasu, P. Godefroid, M. Polishchuk, and B. Ray. Pythia: Grammar-Based Fuzzing of REST APIs with Coverage-guided Feedback and Learning-based Mutations. *arXiv preprint arXiv:2005.11498*, 2020.

[15] V. Atlidakis, P. Godefroid, and M. Polishchuk. RESTler: Stateful REST API Fuzzing. In *Proceedings of the 41st IEEE/ACM International Conference on Software Engineering (ICSE)*, 2019.

[16] V. Atlidakis, P. Godefroid, and M. Polishchuk. Checking Security Properties of Cloud Services REST APIs . In *Proceedings of the 13th IEEE International Conference on Software Testing, Verification and Validation (ICST)*, 2020.

[17] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning Phrase Representations Using RNN Encoder-decoder for Statistical Machine Translation. *arXiv preprint arXiv:1406.1078*, 2014.

[18] R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. University of California, Irvine Irvine, 2000.

[19] GitLab. GitLab. https://about.gitlab.com.

[20] Godefroid, Patrice and Polishchuk, Marina and Atlidakis, Vaggelis. Automatic Intelligent Cloud Service Testing Tool, 2018. US Patent App. 15/992,727.

[21] S. Newman. *Building Microservices: Designing Fine-grained Systems*. O'Reilly Media, Inc., 2015.

[22] I. Sutskever, O. Vinyals, and Q. Le. Sequence to Sequence Learning with Neural Networks. *Advances in NIPS*, 2014.