Poster: Denial of Wallet Preemptive Defence - Attack Simulation and Vulnerability Scouting

Daniel Kelly
School of Computer Science
National University of Ireland, Galway
Galway, Ireland.
Email: d.kelly69@nuigalway.ie

Frank G Glavin
School of Computer Science
National University of Ireland, Galway
Galway, Ireland.
Email: frank.glavin@nuigalway.ie

Enda Barrett
School of Computer Science
National University of Ireland, Galway
Galway, Ireland.
Email: enda.barrett@nuigalway.ie

***Abstract*—The intentional targeting of components in a cloud based application, in order to artificially inflate usage bills, is an issue application owners have faced for many years. This has occurred under many guises, such as: Economic Denial of Sustainability (EDoS), Click Fraud and even secondary effects of Denial of Service (DoS) attacks. With the advent of commercial offerings of serverless computing circa 2015, a potential attack has emerged, termed, Denial of Wallet (DoW). We describe our ongoing development of a safe means to simulate these attacks as well as an initial investigation into the suitability of the current safeguards offered by one of the largest cloud providers, Amazon Web Services (AWS), to combat DoW. We believe that DoW may become ever prevalent as services become further abstracted. Given the nature of the damage caused, such attacks may not be disclosed should they happen. As such, we believe that the development of an attack simulator and specific testing of security measures against this niche attack will be able to provide previously unavailable data and insights for the research community. We have developed a prototype DoW simulator that can replicate months worth of API calls in a matter of hours for ease of data generation. We have also begun testing deployments of serverless applications on AWS for vulnerabilities that may lead to DoW with a view to further refine our simulator. Our aspiration for the future of this work is to provide a framework and starting point for research on this form of attack.**

***Index Terms*—denial-of-wallet, serverless computing, function-as-a-service, attack modelling, attack simulation**

I. Introduction

Serverless computing has emerged as a powerful paradigm for application development. The appeal of decreased time to deployment, lack of servers to manage and the pay-per-use cost model of the functions that execute the business logic has accelerated its adoption by many application owners. These serverless function driven applications are highly scalable, such that a weak attempt at a flooding style DoS attack may be absorbed with no disruption to service. However, such a capability also leaves serverless functions vulnerable to an evolution of the EDoS attack. This evolution has been named "Denial of Wallet" [1]. DoW can describe any abuse of a pay-per-use cloud product to cause an inflated bill. However, in the scope of this research, we will take it to mean the continual triggering of serverless functions in an attempt to induce greater operation costs for the application owner.

As this attack targets the capital of application owners, we devise an environment for calculating varying DoW attempts without incurring real financial losses. Our environment emulates the pricing formulae and structure of the four major commercial serverless platforms; AWS Lambda[1], Google Cloud Functions[2], Microsoft Azure Function[3] and IBM Cloud Functions [4]. Our system also serves as a tool for generating usage and attack data for future training of mitigation systems in the absence of datasets of historical DoW attacks. We model three likely forms of attack; *continuous rate, exponential* and *random rate function* spamming, operating on botnets with varying forms of IP address changing and recycling. These attacks are run against modelled traffic based on AWS' serverless load testing use case [2]. Furthermore, all attacks can be executed in scenarios of bursty and non-bursty traffic.

Further to simulating attacks, we have initiated an investigation into how such attacks may bypass the current safeguards offered by a cloud provider. Specifically, we look into the security of AWS based serverless applications. Following the safety standards outlined in a recent whitepaper on serverless development [3], we created a test application for the purpose of identifying vulnerabilities that may allow for DoW. The findings of such investigation will serve to further refine our simulator.

II. Simulator Design

Denial-of-Wallet Test Simulator (DoWTS), is a simulated serverless platform that will emulate the cost damage of DoW and generate pseudo timestamped datasets of request traffic[5]. DoWTS calculates the current cost of serverless function invocations per simulation time step for the four largest commercial platforms. It also generates usage log data of every function invocation with a label denoting whether it was a bot or legitimate. Such data will be used in future research on classifying legitimate traffic.

DoWTS is composed of three main components; Usage Generator, Serverless Platform Emulator and a MongoDB database.

[1]https://aws.amazon.com/lambda/
[2]https://cloud.google.com/functions
[3]https://azure.microsoft.com/en-us/services/functions/
[4]https://cloud.ibm.com/functions/
[5]https://github.com/psykodan/DoWTS

The Usage Generator takes in a number of parameters from the user that define the number of requests, the time scale and real vs. bot traffic ratio in a given usage scenario. DoWTS is capable of generating thousands of requests per second and therefore can be used to simulate a DDoS attack. It can also be used to perform simulations of long time span attacks, such as leech attacks [1]. The rate of requests can also be randomised for inconsistent timings of attack. DoWTS also implements methods of recycling and changing the IP addresses of the botnet to either emulate a changing botnet or spoofing an IP address. The botnet and real users can also be set to make requests in bursty increments. This method of varying traffic to provide heavy noise in attack scenarios making it a more difficult task to differentiate between real and fictitious traffic.

The traffic generated by the Usage Generator is given a random timestamp within the confines of a time step which is persisted along with other information such as the IP address of the user/bot, function ID and a label of whether it is a bot or not to the database. The result is a dataset of good and bad traffic that can be used in the training of classification algorithms for DoW detection.

To model regular use on a given application, DoWTS uses a Poisson distribution based on a predicted number of users per time step. Serverless applications may experience time dependent load variation [4]. As such, a sine function was used to give variation in the traffic load depending on the time of day. In future, timings of requests may be generated based on usage data recorded by Intrusion Detection Systems (IDS) on existing applications [6].

The Serverless Platform Emulator is configured with memory allocation and function execution time of the theorised functions in an application. Functions may then be grouped into chains that logically execute one after another. These chains contain a parameter that allows for the distribution of traffic to certain chains more than others where there are functions more commonly invoked than others. The Serverless Platform Emulator allows us to compare the effects of DoW across multiple commercial platforms. Utilising the openly available pricing formulae for each platform, the emulator keeps count of the total cost, invocations and runtime of the functions on that platform.

III. Vulnerability Discovery

There are a number of best practices outlined by AWS [3], [5] on how to develop for serverless architecture in a secure manner. However, there is rarely any specific mention of protection from DoW. There exists numerous official sample applications for deployment on AWS[7][8]. Using these as a base along with rigorous adherence to the security best practices, we can begin to specifically search for vulnerabilities that would lead to DoW. These vulnerabilities will then be incorporated into our simulator for safe testing before testing real attacks on a real hosted application.

[6]https://www.kaggle.com/datasets/cicdataset/cicids2017

[7]https://github.com/aws-samples/aws-serverless-workshops

[8]https://aws.amazon.com/lambda/web-apps/

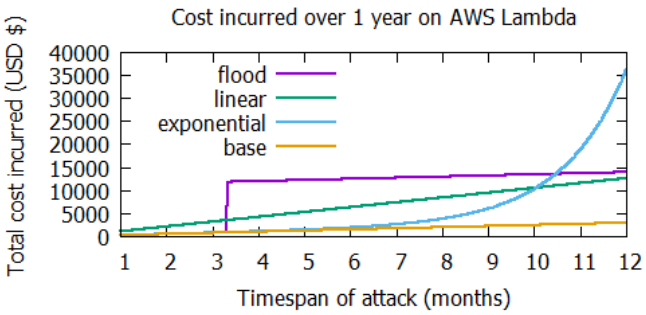

Fig. 1. Cost incurred over 1 year from varying attacks on AWS Lambda

IV. Preliminary Results

The results in Figure 1 show the total costs of the base usage and the damage caused by various attacks on functions hosted on AWS Lambda. The following attacks were chosen based on plausible values. These values are presented for demonstration purposes and further research into plausible attack vectors will be conducted. These attacks do not utilise the IP changing or bursty traffic modes, they simply demonstrate theoretical financial damage that could be caused.

Flooding - HTTP Flood style attack 50,000 requests per second for 12 hours (start time of attack is irrelevant)

Constant Rate Leaching - Leech attack, performing 2000 requests per bot per hour on a botnet of 100 nodes

Exponential Rate Leeching Leech attack, starting at 10 requests per bot per hour on a botnet of 100 nodes. Increasing number of requests every hour by factor of 1.001.

V. Future Work

Future work will involve continued investigation into vulnerabilities on commercial serverless platforms. We aim to compose a list of our findings across the major serverless platforms. Future work utilising the data generated by our simulator will encompass the training of detection algorithms and development of mitigation systems that will be deployed first to serverless platform simulators that function in real time [1], then later to real commercial platforms. Such work will advance knowledge on serverless computing security and DoW prevention.

References

[1] D. Kelly, F. G. Glavin, and E. Barrett, "Denial of wallet—defining a looming threat to serverless computing," *Journal of Information Security and Applications*, vol. 60, p. 102843, 2021.

[2] J. Beswick, "Load testing a web application's serverless backend," 2020. [Online]. Available: https://aws.amazon.com/blogs/compute/load-testing-a-web-applications-serverless-backend/,urldate=2021-10-20

[3] AWS, "Serverless applications lens aws well-architected framework," Report, 2019.

[4] D. Kelly, G. Frank, and B. Enda, "Serverless computing: Behind the scenes of major platforms." IEEE, 2020, Conference Paper, pp. 304–312.

[5] AWS, "Security overview of aws lambda," Report, 2021.