CS 6480: Paper reading summary
HA 3.a

José Monterroso

School of Computing, University of Utah

September 1, 2020

1 Title of paper: eZTrust: Network-Independent Zero-Trust Perimeterization for Microservices

Paper discussed in this summary is "eZTrust: Network-Independent Zero-Trust Perimeterization for Microservices" [3].

1.1 First pass information

1. *Category:*

   This paper is a description of a research prototype. The prototype being discussed in this paper is eZTrust, a network-independent perimeterization approach for microservices.

2. *Context:*

   The technical area of this paper is internal network security with regards to microservices. We have seen similar concepts being discussed in "Containerization and the PaaS Cloud" [2]. These concepts are containerization, lightweight containers, lifecycles of containers, and microservices.

3. *Assumptions:*

   One big assumption is that the authors assume that internal security is a huge risk. This is the whole premise of their argument. I do think this is a valid assumptions, and the authors provide papers to back up their claims.

4. *Contributions:*

   The main contributions are as listed. First, they design the eZTrust architecture which enables fine-grained context based perimeterization. Second, they implement a proof-of-concept prototype of the architecture. And lastly, they quantify the performance and resource overhead of the prototype and compare it against alternative approaches.

5. *Clarity:*

   Yes, I do think the paper is well written. I do enjoy the way it's formatted, and so far it feels like an interesting read with great claims that are backed up by other papers.

1.2 Second pass information

- *Summary:*

  eZTrust is a policy driven perimeterization access control system for a containerized microservices environment. eZTrust's goal is to contain the attacker from moving laterally from one compromised workload or application to another [3]. This is done through the use of eBPF-based tracing mechanisms to monitor various microservice-related events. As well as probing via harvester, which collects additional runtime environment

related contexts of microservices. eZTrust is intrinsically more scalable then network-endpoint-based perimeterization because packets are classified into flows based on microservice contexts. The eZTrust prototype is implemented in Python/C and is integrated with the Docker runtime environment. Furthermore, the paper provides multiple uses cases that are improved thanks to the use of eZTrust. Lastly, eZTrust is evaluated on the complexity of its overhead, correctness in dynamic environments, and compared against alternative schemes in terms of performance and overhead.

1.3 Third pass information

- *Strengths:*

  This paper in-general is very detailed, which I like because I feel like I can reimplement their work. I considered the related works to be a strength because not only did it provide us with background information, but the authors presented it in such a way that gave the reader an idea that work still needs to be done in this area, ergo, legitimizing their own work. The flow charts help with understanding the logic while the graphs present latency, and CPU utilization in a readable way. The Motivational Use Cases section is huge strength in favor of the paper because it applies real world applications of how eZTrust can help. Furthermore, the Evaluation section offers a lot of insight into the performance of eZTrust. The evaluation is a strength because not only does it give you a detailed overview of the set up, but it helps the readers understand the abilities of eZTrust.

- *Weaknesses:*

  A weakness I found was the introduction. I personally felt that it gave me way to much information. I would have almost liked to have seen half of the information from the introduction into a background or approach section later in the paper.

- *Questions:*

  One question I had was why must we use a different eBPF program for ingress and egress packet processing. I also didn't really understand a lot of the architecture design ideas because I have no prior knowledge of perimeterization.

- *Interesting citations:*

  I'm not very familiar with microservices. My knowledge of microservices began at week two of this course, so I would really like to read "Building Microservices" [1].

- *Possible improvements:*

  There was just that information in the introduction that I think should have been moved to the Architectural Design section, or could have been added to an "Approach/Background" section.

- *Future work:*

  I'm very new to the world of perimeterization, so I think any type of research into perimeterization would be interesting. Furthermore, the paper lists motivational use cases like vulnerability-driven perimeterization, control flow integrity and user identity based firewall, each of which could be potential areas of study in the future. It is because this paper is very detailed I think we could reimplement eZTrust and potentially try to make it better by applying it to more use cases.

References

[1] Newman, S. In *Building Microservices: Designing Fine-Grained Systems* (2015), O'Reilly.

[2] Pahl, C. Containerization and the paas cloud. *IEEE Cloud Computing 2* (July 2015), 24–31.

[3] Zaheer, Z., Chang, H., Mukherjee, S., and Van der Merwe, J. eztrust: Network-independent zero-trust perimeterization for microservices. *SOSR '19: Proceedings of the 2019 ACM Symposium on SDN Research* (Apr. 2019), 49–61.