----------------------- REVIEW 1 ---------------------

SUBMISSION: 7992

TITLE: On Detecting Malicious Code Injection by Monitoring Multi-level Container Activities

AUTHORS: Md. Olid Hasan Bhuiyan, Souvik Das, Suryadipta Majumdar and Md. Shohrab Hossain

----------- Overall evaluation -----------

SCORE: 1 (weak accept)

----- TEXT:

The paper presents an approach to detect malicious code injection attacks in Docker containers. Their approach is based on monitoring logs, and system calls to identify the differences between benign and malicious attacks. The authors simulate their approach on three containers (a built-in website and two containers of SQLi)

Strengths:

\* The paper is well written.

\* The topic of the paper on security issues is a relevant topic for the community.

\* The proposed approach is a simple approach based on analyzing logs.

Weaknesses:

\* I wonder if the approach can easily distinguish between a high workload and an attack. For a high workload, the frequency of an event will be higher, and the sequence of events can change the same way as an attack.

>> This gives you an idea to run some new experiments. Now what you need to find is how you vary the workload and then put various workload as x-axis to measure its effect on your approach

\* Some thresholds are not well justified. An example is the sliding window of the sequence of events between 2 and 10. Another example is the threshold 82%; if 82% of the system calls exceed a certain usual frequency criterion, the sequence is considered an attack.

>> you need to justify the choice of the threshold. Additionally, you have to discuss how other threshold choices might affect the results.

>>Bring in some more charts for other percentage

\* The evaluation of the proposed approach is missing too many details. For example, how different thresholds (e.g., the upper and lower bounds) are measured? Are they defined on the same data for training and evaluation? How are the training and evaluation data obtained? How precision and recall are measured? Is precision when the approach identifies that a container has a security issue? If so, the precision is either 0 or 100%? The same question for Recall.

>> By answering each of these questions, you can improve the clarity of your experiment section.

\* From a practical point of view, the authors did not evaluate or discuss how their approach would be used in practice. Is it to detect a malicious code before it happens, so practitioners should first simulate an attack and a benign execution? Is the approach to be used when a container is running in production? I do not believe so since the approach requires the execution of an attack to define multiple thresholds that are required for their approach. Is it a general approach that can fit any container? Still, I do not believe so, as logs can be different from one container to another.

>> you may want to DD discussion section right before conclusion; where we add a discussion to show how your solution can be adopted in practice mainly by answering the questions the reviewer is asking here.

----------------------- REVIEW 2 ---------------------

SUBMISSION: 7992

TITLE: On Detecting Malicious Code Injection by Monitoring Multi-level Container Activities

AUTHORS: Md. Olid Hasan Bhuiyan, Souvik Das, Suryadipta Majumdar and Md. Shohrab Hossain

----------- Overall evaluation -----------

SCORE: -1 (weak reject)

----- TEXT:

Summary:

  This paper addresses the security concerns associated with code injection in containerized environments by proposing a multi-level monitoring-based detection approach.

  The proposed approach monitors container activities at system call level and orchestrator level, enabling the detection of malicious code injections by identifying deviations from expected behavior.

  The authors implemented and evaluated the approach specifically for Kubernetes, and selected a limited set of system calls to monitor.

Strengths:

  - This subject is important nowadays due to the growing reliance on cloud-native applications and containerization,

    which necessitates effective measures to detect and mitigate the security risks posed by malicious code injection attacks.

  - Proposes various approaches for detecting malicious code injection in a containerized environment

  - The authors performed extensive experiments to find optimal set of system calls to use in order to get the best overall performance.

Weakness:

  - The paper does not provide comparable results with existing literature. These results would provide a more comprehensive understanding of the strengths and weaknesses of the proposed approach.

>> if you can add any comparative experiment that significantly add value to your wor. Not necessarily you need to reimplement any existing methods, you can also simply implement a fictional (but reasonable) naïve solution and compare it with yours.

>>Comparison with Performance Evaluation && Taking a Peek.

  - Paper lacks comprehensive analysis of how implementing the proposed approach in real-world production environments could affect system performance or resource utilization.

>> add another point to the new discussion section to address this point. Mainly by description say how your solution can tackle this scenario.

  - The approach depends on predetermined attack scenarios, the effectiveness would be limited in detecting new emerging attack methods.

>> in the discussion section, add another point to say how your solution can be adaptive to emerging attack methods.

  - The literature review is not very strong, and references contains only ~15 articles out of 34 are peer-reviewed in the references.

>> see if more decent references (e.g., good conferences and journals – I can check once you send a list of references) can be added.

Minors:

  - Some references are not related to the paragraph they are inserted into (e.g [25], [14], [15])

  - Some usage of the word "Virtualization" confused me, I would prefer "Containerization" or "OS Level Virtualization".

    To not confuse it with full-virtualization with an hypervisor.

>> Fix all those minor issues.

>> Can be easily fixed

----------------------- REVIEW 3 ---------------------

SUBMISSION: 7992

TITLE: On Detecting Malicious Code Injection by Monitoring Multi-level Container Activities

AUTHORS: Md. Olid Hasan Bhuiyan, Souvik Das, Suryadipta Majumdar and Md. Shohrab Hossain

----------- Overall evaluation -----------

SCORE: -1 (weak reject)

----- TEXT:

Containers are one of the most popular technologies used in the current IT field. Code injection attacks in containers, consequently, is an important problem to solve. The authors present a scheme to detect malicious attacks.

I feel that the authors have implemented an interesting solution. But I have three main concerns with the paper.

(1) The paper does not clearly bring out what are the main scientific contributions of the paper. The seems to be a fair amount of engineering involved. But what is the novelty - it is unclear.

>> In the intro, we have to mention the problem in a better way to show how our multi-level approach is useful and novel. If you can clearly state the research challenges in this problem, then this worry might be reduced.

(2) The paper is a bit difficult to read, especially for someone like me, who is an outsider.

(3) Evaluation (Section 5) looks too brief and some comparison with prior work would have been better.s

----------------------- REVIEW 4 ---------------------

SUBMISSION: 7992

TITLE: On Detecting Malicious Code Injection by Monitoring Multi-level Container Activities

AUTHORS: Md. Olid Hasan Bhuiyan, Souvik Das, Suryadipta Majumdar and Md. Shohrab Hossain

----------- Overall evaluation -----------

SCORE: -1 (weak reject)

----- TEXT:

Summary.

--------------

This paper proposes an approach to detect attacks against containerized systems. The approach first defines benign behavior and then simulate attacks against a container, where the simulated behavior can divert from the benign behavior. Experimental results show that the approach can detect malicious behaviour with high accuracy.

Strengths.

--------------

1. Practical problem to address and easy-to-read paper

2. High accuracy of the approach

Weakness.

--------------

1. Details of a benign behavior and how it differs from a malicious behavior is missing, as such the core contribution of modeling benign behaviour seems weak

2. The simulation approach seems ad-hoc and manual, which could be a threat to generalizability

Detailed Review.

--------------

This paper is generally well-written and easy to understand. The problem is practical. However, the paper is a bit premature at its current stage. It has many missing details and the evaluation seems almost non-existent. I think the paper could benefit from addressing the following comments.

It is not clear what constitutes a benign behaviour and how it differs from a malicious behavior during the modeling process. Would the sequence of system calls differ between benign and malicious behaviors? If so, what would be an example of a system call sequence with a malicious behavior?

>> you must add these details in Sec 2 and/or Sec 3.

Fig 3 offers visuals on the proposed approach. Very little of figure 3 is explained in the texts of the paper. Unfortunately, it’s hard to understand the approach by simply looking at Fig 3. For example, how do you detect a benign call and filter it from? Is it a signature-based approach, i.e., you only check a list of system call sequences and if a sequence does not match those in the list it is a malicious call? If so, how do you ensure the robustness and completeness of those signatures? None of these are addressed.

>> Figure 3 has to be redone and also its texts (a high quality approach overview figure can decide the fate of your paper). You are not doing justice to your work with this figure. I suggest you involve Shafayat in the next submission and ask his help to redesign this figure where we want to highlight the contribution of this proposed work while keeping it simple and easy-to-understand. Additionally, address all those questions above about this figure and its description.

The approach is validated by using simulated attacks. This is not problem if the simulation is done by following standard process. The paper notes in Section 3.3 “From the attack path provided by CNCF, four different attack paths can be identified that finally lead to malicious code injection.” What are these paths? Why defined those paths? So, if we simply create rules to detect those paths, we could mitigate those attacks? If so, why do we need to model benign behavior?

>> You need to clearly describe and justify that your simulation follows standard process. Now it is sounding too arbitrary and hence the reviewer has this comment. You need to address his questions.