**Problem Statement**

Cybersecurity has been a dynamically growing and evolving field for the past 50 years [1]. As technology continues to advance, it is important to remain abreast with the new technologies and software development tools being deployed to support these new advancements and keep users safe. In cybersecurity, one of the key important activities is development operations which relies on automation and software tools. This project will investigate Docker and highlight the security risks it faces.

**Why is it Useful?**

Cybersecurity tools are the different applications or software programs companies use to protect their systems and networks from unauthorized access [2]. These software development tools are important as unwanted access to a company’s system can lead to cyberattacks, identify theft or data theft causing significant issues for organizations and their affected customers [2]. Looking into current security issues faced by these software tools and discovering new issues would bring the necessary concerns and threats to the forefront and make the right people aware of them.

**Case Study – Docker Security Risks**

Docker is a software tool that automates the deployment of applications inside containers by launching Docker images [3]. Docker allows applications to run independently of one another without affecting each other or the host machine. For example, Amazon ECS allows users to run applications on a managed cluster of Amazon EC2 instances in Docker containers [3]. Due to Docker’s growing popularity, it is predicted that more than 70% of global companies will have more than two containerized applications in production by 2023 [4]. This move to more containerized infrastructure is due to the advancement in technology, especially network and 5G technology [4].

For general uses, millions of Docker images have been downloaded from Docker Hub by users for website deployment, data management and personal and business tasks [3]. The downside to Docker’s popularity is that it makes it a desired target for attackers. Due to this alone, it should be a top priority of developers and cybersecurity engineers to ensure Docker is as secure as possible in an effort to keep customer information and proprietor information safe. However, it is hard to perform a comprehensive analysis into Docker security issues for a variety of reasons: (1) Security impacts of some categories of information on Docker Hub are ambiguous [3]. An example is the use of run commands. Run commands are special instructions that are critical to the security of Docker containers [3]. Although these run commands are critical to Docker security, there is still no method available for measuring the security risks of using run commands [3]. This could prove to be quite dangerous as run commands are being used to ensure the security of docker containers, but the security of run commands are unknown. Security analysis of run commands requires significant empirical analysis and manual effort [3]. (2) Obtaining and analyzing Docker images and the associated meta information in a scalable manner is significant [3]. Performing a complete analysis on Docker images is difficult since they contain many files in a range of types [3].

From these challenges, the study by *Liu, P. et al.* has identified 3 major sources of security risks in Docker Hub, namely sensitive parameters in run commands, malicious docker images and unpatched vulnerabilities [3].

**Threat model used in study (Liu, P. et al.)**

The study identified two main attack models to base their research and results on. Firstly, they considered the attacker attacking the vulnerabilities in the images and meta information the developer uploads to the docker hub. If an attacker identifies these vulnerabilities, any user who downloads and runs these images is susceptible to attack. The run commands used to run these images may also contain sensitive parameters. For example, omitting a specific word or adding a specific word in run commands can have catastrophic results for the host and user. Secondly, the attacker can upload malicious images and run commands within the hub container containing safe run commands and images approved by the developers. The average user would not be aware of these malicious images or commands and once they are downloaded or used the user can suffer attacks such as crypto mining [3].

**Data Collection and Analysis**

*Liu, P. et al.* collected a wide range of Docker images from the Docker Hub along with public data of those images such as name, repository description and developer information [3]. The random collection of data and images would result in a less biased and more reliable data set. It would be counter intuitive to search for images y suspect to be compromised as an attacker is free to place a compromised image where they see fit.

An extractor was then used on the data to obtain run commands and sensitive parameters related to the data set which was then analyzed for security purposes [3]. The analysis of the data was done by security analysis tools, which means that the results collected are only as accurate as the accuracy/efficiency of the tools identifying these vulnerabilities. Reporting the accuracy/efficiency of these tools would help give the reader an estimate/error margin to compare to and help them account for an expected number of false negatives and positives. A study by *Wisk, K. et al.* also raised these concerns and suggested targeting false negatives and positives in container scanners by integrating machine learning.

In an attempt to mitigate these discrepancies in results, one malware analysis tool called VirusTotal was used. Out of 36,584 uniquely executed programs, only 13 malicious programs appeared in 17 images [3]. It is good to note that none of these malicious programs were found in official repositories but in community repositories. This is more than likely due to the fat that hackers are more likely to prey on community individuals who are not knowledgeable.

The study by *Wisk, K. et al* also performed extensive vulnerability analysis on 2500 Docker images and found similar results. This study found that not only are the number of newly introduced vulnerabilities on Docker increasing but official images are the least vulnerable to attack while community images were the most likely to be attacked. The study showed that out of the top 10 most vulnerable images, there were 8 community images, 1 official image, and 1 verified image. This study also went a step further and showed that the most vulnerabilities originated from two popular scripting languages, namely JavaScript and Python, with Python 2.x packages and Jackson-databind packages containing the highest number of severe vulnerabilities [4]. This raises the concern that due to the interconnectivity of the DevOps process, the vulnerabilities of other software development tools can infringe on the security of Docker and that scripting languages should implement encryption protocols and proper coding practices at all times.

**Table 1:** Comparison between four studies and their results for the security issues of different Docker repositories.

Table

Description automatically generated

*Table taken from [7].*

A comparison table was also used to confirm the results found in both papers mentioned above. A study done by *J. Gummaraju, et al.* in 2015, *R. Shu et al.* in 2017 and *E. Socchi et al.* in 2019 confirmed the results of study [4] by proving that not only was Docker facing security issues but the official repositories in Docker were generally less likely to be affected by attackers compared to community repositories.

In future studies, it would be interesting to see if different types of Docker images/containers are more likely to be attacked than others. For example, those that contain more sensitive information or data which attackers could use for harm.

**References:**

1. C. Townsened, “A brief and incomplete history of Cybersecurity,” *United States Cybersecurity Magazine*, 02-Nov-2020. [Online]. Available: https://www.uscybersecurity.net/history/. [Accessed: 10-Sep-2022].
2. Indeed Editorial Team, “Indeed career guide | indeed.com Australia,” *Cybersecurity Tools: Definition, Importance and Examples*, 02-May-2022. [Online]. Available: https://au.indeed.com/career-advice/career-development/cybersecurity-tools. [Accessed:10-Sep-2022].
3. Liu, P. *et al.* (2020). Understanding the Security Risks of Docker Hub. In: Chen, L., Li, N., Liang, K., Schneider, S. (eds) Computer Security – ESORICS 2020. ESORICS 2020. Lecture Notes in Computer Science(), vol 12308. Springer, Cham. https://doi.org/10.1007/978-3-030-58951-6\_13
4. Wist, K., Helsem, M., Gligoroski, D. (2021). Vulnerability Analysis of 2500 Docker Hub Images. In: Daimi, K., Arabnia, H.R., Deligiannidis, L., Hwang, MS., Tinetti, F.G. (eds) Advances in Security, Networks, and Internet of Things. Transactions on Computational Science and Computational Intelligence. Springer, Cham. <https://doi.org/10.1007/978-3-030-71017-0_22>
5. J. Gummaraju, T. Desikan, Y. Turner, Over 30% of official images in Docker hub contain high priority security vulnerabilities. Tech. Report, Banyan Ops, 2015
6. R. Shu, X. Gu, W. Enck, A study of security vulnerabilities on Docker Hub, in *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy* (2017)
7. E. Socchi, J. Luu, A deep dive into Docker Hub’s security landscape - a story of inheritance? (2019). <https://tinyurl.com/y8axga4g>.