Compass

Release 1.0.0

Joe Samuel

Apr 18, 2022

OVERVIEW

Compass is a toolkit to support system architects, developers, evaluators, and researchers to design secure technology systems.

CHAPTER

ONE

ABOUT

System architects, developers, and evaluators need simple and intuitive tools to help them design secure software systems. While a number of secure system design tools exist, it is hard for system architects and other stakeholders to keep up with the availability and improvement of these tools over time. This problem is compounded as more secure system design tools are released.

To tackle these challenges, we developed Compass, a web-based toolkit to house secure system design tools. The goal with Compass is to create a one-stop-shop for secure system design tools to help system architects and other stakeholders keep up with the different tools and techniques to design secure systems.

With Compass, we also provide design and usability guidelines to help researchers and develop tools that are intuitive for system architects to use. We believe this toolkit will inspire more secure system design researchers to develop tools for practitioners to apply, try out, and improve. This aims to address the lack of adequate tool support for practitioners to apply secure system design research in their workflows.

Note: Compass toolkit is discussed in greater depth in [CARLETON2021] and [QRS2021].

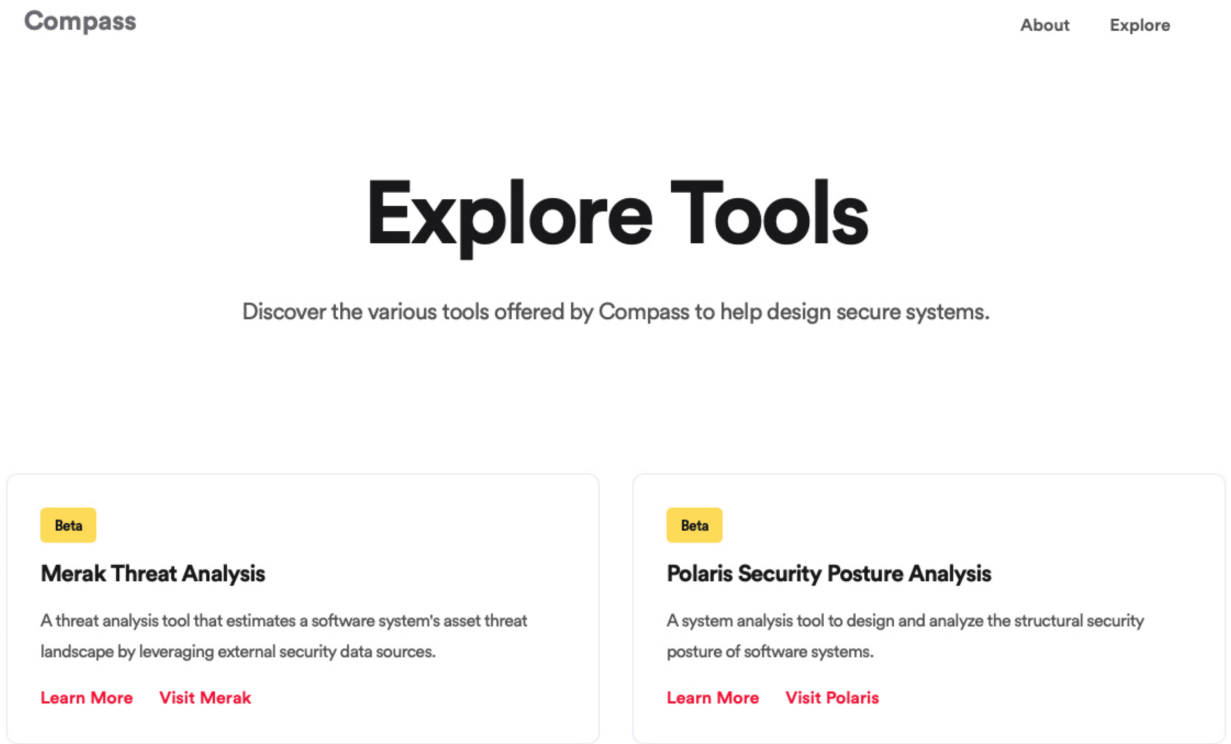
CHAPTER

TWO

CONTRIBUTE

Developing tools to be deployed on Compass is fairly straightforward. Developers have the freedom to select any technology stack that allows them to deploy their tool as a cloud-native web application. This allows each tool to have independent code repositories and development pipelines and contribution policies (such as being open-source).

Because we use Docker as our container technology, we, therefore, require tools to be containerized using Docker to enable us to deploy and manage the application using Compass. The image for the containerized application needs to be stored in a container registry accessible to Compass which takes care of deploying and managing the access and availability of the tool. Tools deployed on Compass will be showcased on the Explore page as shown below.



If you are interested in proposing a new tool or modification to an existing tool at Compass, please get in touch with us.

CHAPTER

THREE

GLOSSARY

Secure Sockets Layer A protocol to secure communication between networked computers.

CHAPTER

FOUR

REFERENCES

CHAPTER

FIVE

REQUIREMENTS

With Compass, we aim to abstract the challenges associated with deploying and maintaining web-based secure system design tools. Such challenges include scaling tools based on usage, handling security-related concerns such as establishing secure connections between the tools and users, managing the availability of tools, and more. By abstracting these concerns, we seek to enable researchers and developers to primarily focus their efforts on the functionality and intuitiveness of their tools rather than the logistics of deploying a web-based application.

However, we do have a few requirements for a tool to be made available through Compass:

- Solve a security-related problem
- Be web-based (API, GUI, or Both)
- Be cloud-native (containerized using Docker)

CHAPTER

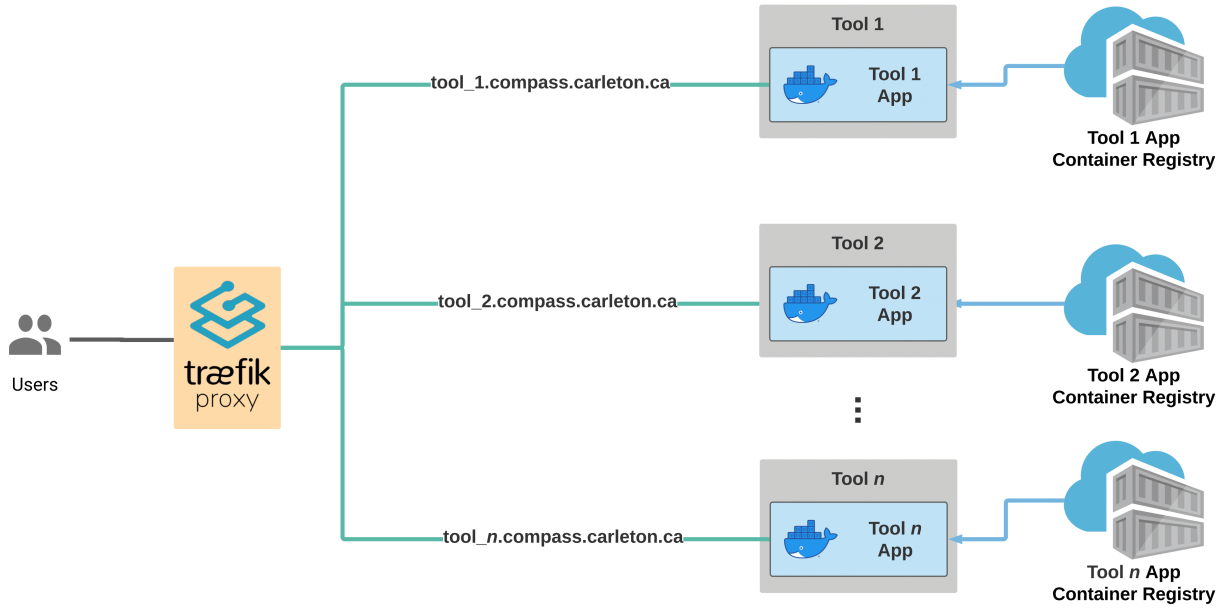
SIX

BUILD

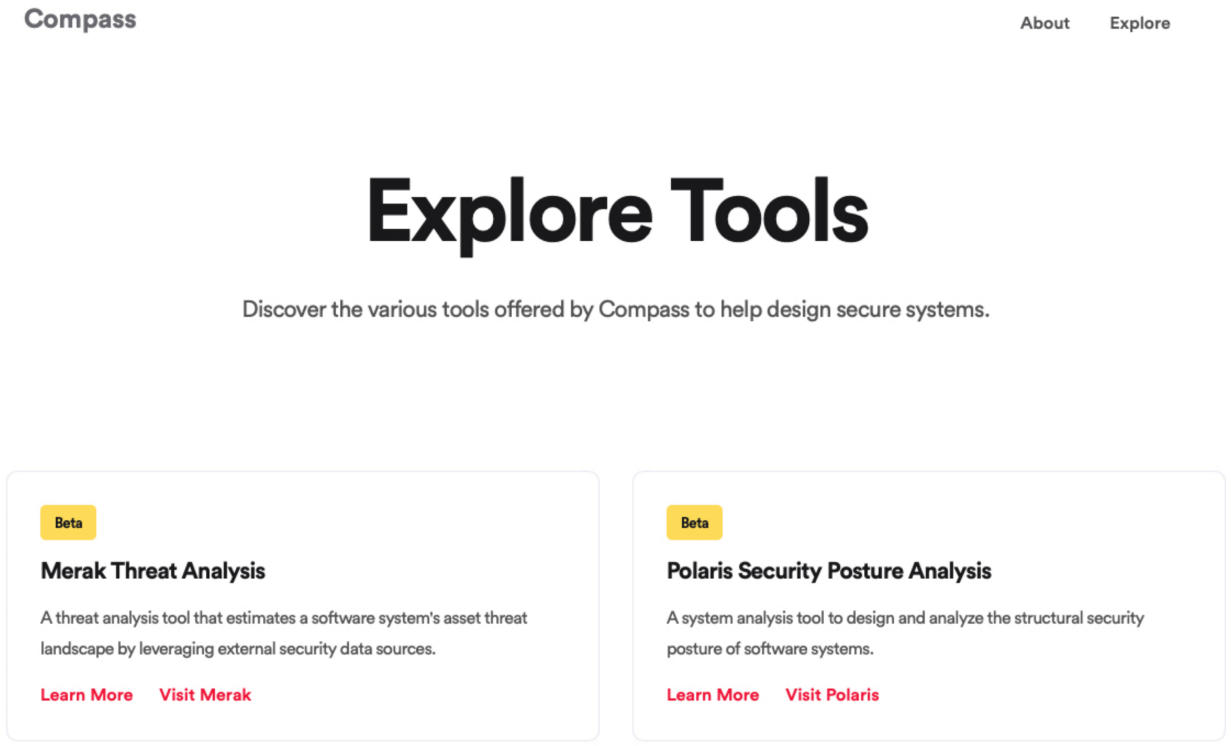
Developing tools to be deployed on Compass is fairly straightforward. Developers have the freedom to select any technology stack that allows them to deploy their tool as a cloud-native web application. This allows each tool to have independent code repositories and development pipelines and contribution policies (such as being open-source). This versatility is achieved by the requirement for your application to be containerized for deployment.

6.1 Containerize your application

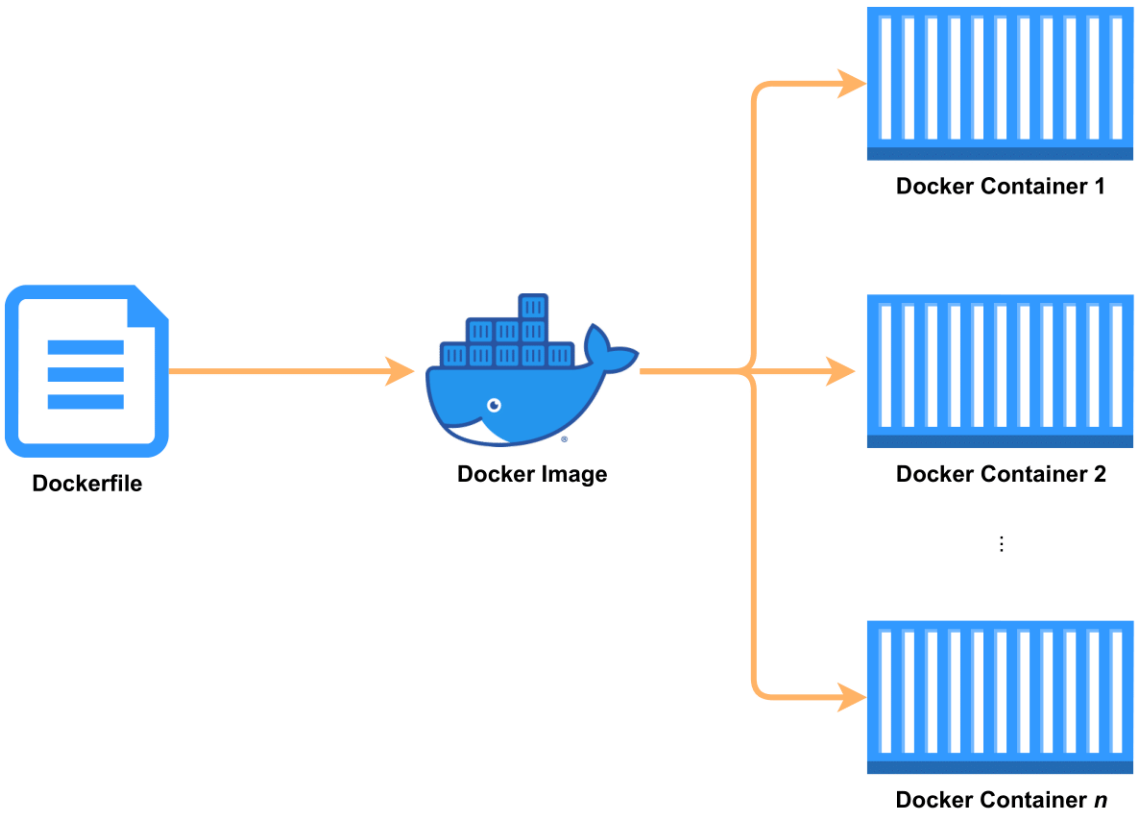
Containerization refers to the packaging of a software application's code along with an operating system and associated dependencies into a lightweight executable called a container [IEEE2014]. Containers provide isolation from the host environment and enable the application to perform in a predictable manner since it was developed and deployed within the same [container] environment. Containers help in standardizing different types of applications to run on any supported infrastructure (one that can run that particular container technology). Containers are saved as container images and those images are stored in container registries. To instantiate the application, the container image needs to be pulled from the container registry and the appropriate container technology can be used to create one or more containers (instances of the application) based on the container image. This allows an application to be replicated, instantiated, and managed independently which helps in maintaining high availability and resiliency. For Compass, we use Docker as the container technology to create and deploy containers [SIGOPS2015]. We chose Docker as it is a popular container technology that is commonly used to develop cloud-native applications. We illustrate the Docker containers associated with the different tools in Compass as blue rectangles (with the Docker logo) in the figure below.



Because we use Docker as our container technology, we, therefore, require tools to be containerized using Docker to enable us to deploy and manage the application using Compass. The image for the containerized application needs to be stored in a container registry accessible to Compass which takes care of deploying and managing the access and availability of the tool. Tools deployed on Compass will be shown on the Compass website as shown below.



Containerizing your application using Docker starts with a Dockerfile. A dockerfile encapsulates the various requirements and sequence of steps necessary to get your application running. Dockerfile varies based on the application stack and you will be able to find Dockerfile templates for your technology stack. To learn more about creating a Dockerfile for your application, refer to Best Practices for Dockerfiles.



Once you configure a Dockerfile, you can run the application locally through Docker to ensure the application works as intended.

6.2 Setting up container registry

Now that you have a working [Docker] container specification (aka. Dockerfile), we can generate a container image based on the Dockerfile. This container image will be used to generate one or more instances (docker containers) of your application. We therefore need to host this container image somewhere accessible so Compass can download and deploy your container image(s) as one or more application instances.

Note: Gitlab, the preferred code hosting platform for Compass Toolkit, provides a free container registry where your application container image can be hosted.

6.3 Setting up CI/CD

We can automate the process of generating and storing container image(s) of your application in the container registry using continous integration and deployment (CI/CD). CI/CD provides the ability to automate the process of building, running, and testing your application in its containerized form. To learn about how you can setup a CI/CD pipeline for your application, refer to Best practices for using Docker Hub for CI/CD.

6.4 Documentation

Documentation serves as a great reference for users to understand and use your tool effectively. There are many methods to offer your documentation. We recommend making your documentation available in the following url: <your_tool_url>/docs. This url format aligns your tool's access to documentation with other Compass tools.

Note: For open-source tools, we recommend readthedocs.io as a method to offer your documentation in a versatile, version-controlled manner where the documentation evolves with your code.

CHAPTER

SEVEN

DOMAINS

We offer custom subdomains for Compass tools under compass.carleton.ca. The format of subdomains will be *<tool_name>.compass.carleton.ca*. The subdomain will be generated and maintained by Carleton University's Information Technology Services (ITS). A custom Secure Sockets Layer (SSL) certificate will also be generated for the tool as part of the Compass infrastructure.

BIBLIOGRAPHY

[CARLETON2021] Samuel, J. F. (2021). A Data-Driven Approach to Evaluate the Security of System Designs (Doctoral dissertation, Carleton University).

[IEEE2014] Dua, R., Raja, A. R., & Kakadia, D. (2014, March). Virtualization vs containerization to support paas. In 2014 IEEE International Conference on Cloud Engineering (pp. 610-614). IEEE.

[SIGOPS2015] Boettiger, C. (2015). An introduction to Docker for reproducible research. ACM SIGOPS Operating Systems Review, 49(1), 71-79.

[QRS2021] Samuel, J., Jaskolka, J., & Yee, G. O. (2021, December). Analyzing Structural Security Posture to Evaluate System Design Decisions. In 2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS) (pp. 8-17). IEEE.

INDEX

S