Runtime Threat Detection

Kubernetes Security

Soldera Marco – s338823

Stella Francesca – s343411

Falco and Tracee

Open Source

Real-Time

eBPF based

Included Rules:

- YAML for Falco
- Rego, Go and signatures for Tracee

For Containarized Systems

For Linux

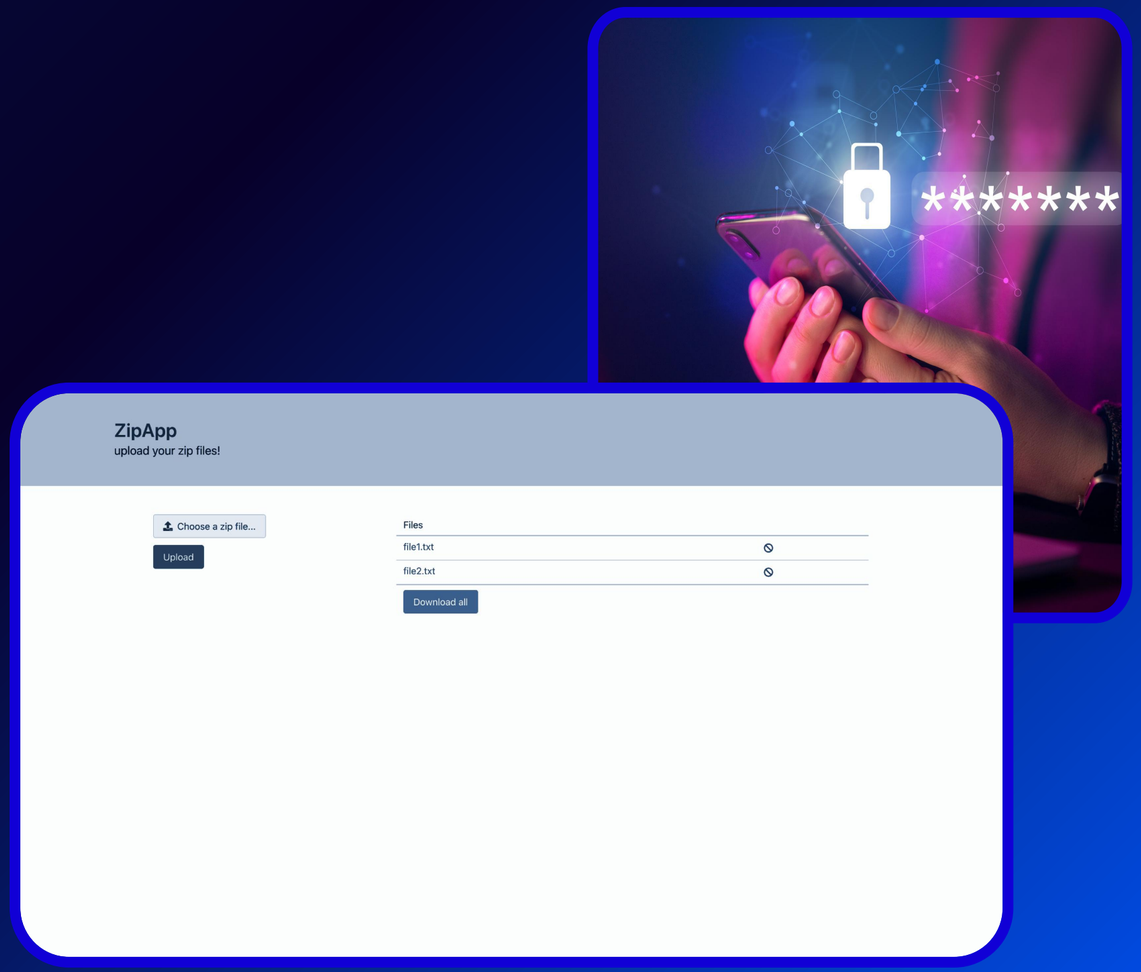
Tools and Scenario Installation

The requirements are the following:

- **Minikube** as local environment or a Kubernetes cluster;
- **kubectl** to interact with Kubernetes;
- **Helm** for package management;
- **Docker** for building and managing container images;
- virtualization engine depending on the OS.

Scenario: ZipApp

We implemented a web application called ZipApp in order to simulate realistic attack scenarios and test our runtime threat detection tools. This web application is intentionally insecure and its key vulnerability is its improper handling of uploaded file names and paths by the “zip” tool.



Event Handlers

We implemented Python servers for Falco and Tracee that are able to apply automatic actions to the deployments within the cluster.
We created a service account for the handler and we associated it to a role with capabilities on pods and deployments of the entire cluster

Attacks in the Scenario

Dropped Executable

Type of attack that involves executing arbitrary executables on the host operating system through a vulnerable application

Reverse shell

Technique that allows an attacker to access a remote computer by initiating a shell session from the target system sometimes bypassing firewall restrictions

ptrace system call

Anti-debugging technique used by a possible malware to detect if it is executed within a debugger

Countermeasures and Their Usage

We analyzed two tools that are provided with Kubernetes.

Pod Security Admission

Set of requirements to be admitted in a namespace with a defined flag that indicates the security level.

Seccomp

Policies to block or allow execution of a set of system calls.

Thank You

Available for Questions

Soldera Marco – s338823

Stella Francesca – s343411