

**Title**

Cloud-Based Threat Intelligence Platform

**Team Members Information**

* **Sarthak Gupta** (1RV22CY051)
* **Adarsh Shrivastava** (1RV22CY004)

**Mentors**

* Dr. Mohana
* Dr. Ramakanth Kumar P, CCVR, RVCE

**Introduction**

This project aims to design and implement a comprehensive, scalable, and secure threat intelligence sharing platform tailored for organizations of all sizes, with flexible deployment options including on-premise, cloud-based, or Software-as-a-Service (SaaS) solutions. The platform will leverage cutting-edge technologies such as artificial intelligence, machine learning, and big data analytics to collect, process, and disseminate threat intelligence in real time. By integrating automated data enrichment, predictive threat modeling, and secure collaboration features, the system will enhance situational awareness and proactive defense mechanisms. Furthermore, it will provide detailed, actionable insights and recommendations for identified threats, enabling security teams to swiftly mitigate risks, strengthen cyber resilience, and adapt to evolving attack landscapes. Through its intuitive interface and robust architecture, the platform will foster seamless cooperation among organizations, industry partners, and government agencies, promoting a unified approach to cybersecurity threat management.

**Literature Survey**

1. **Container Security in Cloud Environments** (Survey 1):
   * Key risks: Shared vulnerabilities in the kernel, outdated images, and poorly configured orchestration platforms (e.g., Kubernetes).
   * Solutions: Multi-layered security architectures, scan during runtime, and during runtime, monitor.
2. **Cybersecurity in Containerization Platforms** (Survey 2):
   * Challenges: Poor authentication, raising privileges, and insecure run settings.
   * Best Practices: Secure sources of images, segmentation of networks, and zero-trust access controls.

**Objectives**

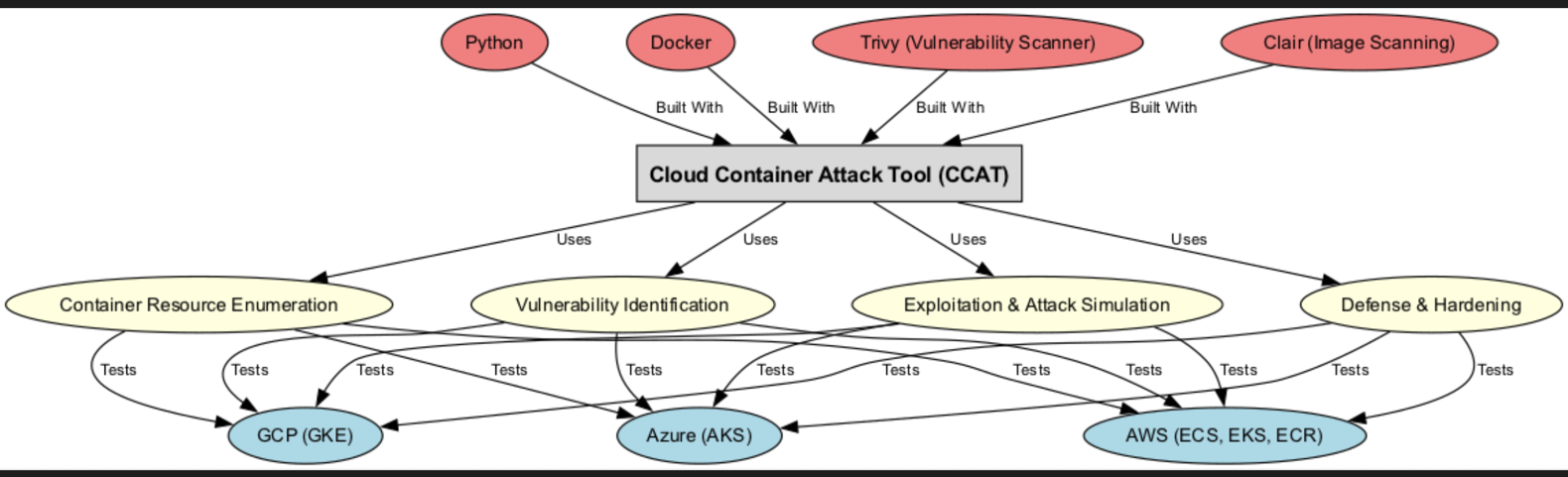
1. Design an in-depth **testing tool** for containerized environments.
2. Simulate **attack scenarios** in real life (e.g., credential theft).
3. Provide **multi-cloud platforms** (AWS, GCP).
4. Serve as an **educational tool** for security training and awareness.

**Methodology**

1. **Enumerate Resources**: Discover containers, registries, and configurations using APIs/CLI tools.
2. **Identify Vulnerabilities**: Use Trivy/Clair for scanning misconfigurations and outdated dependencies.
3. **Exploit Images**: Inject malicious payloads (e.g., reverse shells) into Docker/Kubernetes environments.
4. **Test Detection**: Push modified images to registries (e.g., AWS ECR) to evaluate pipeline security.
5. **Harden Configurations**: Apply patches, restrict permissions, and enforce least-privilege policies.

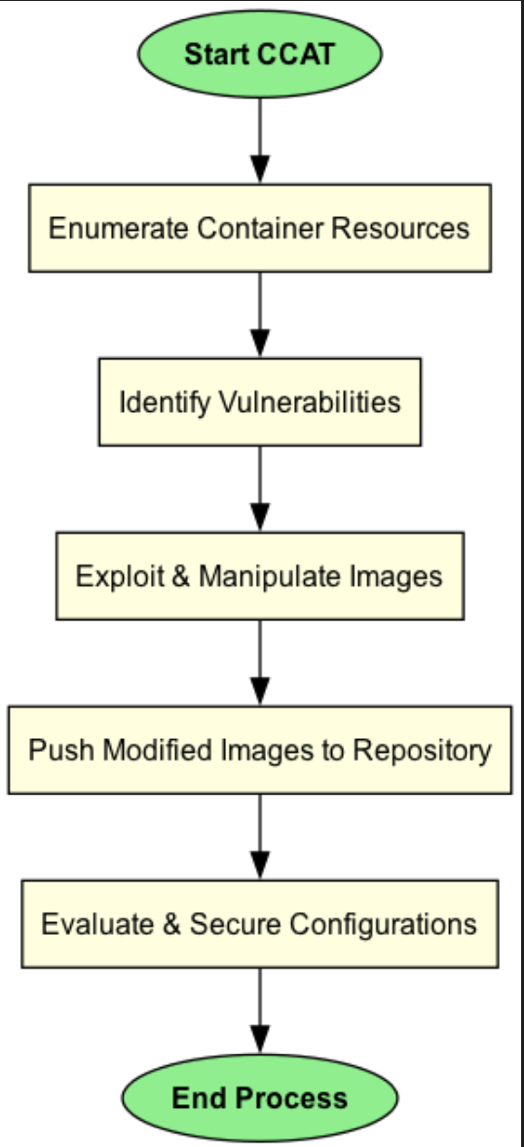
**Design and Implementation**

* **Tools**: Python (core language), Docker (v19.03.1), AWS ECS/EKS, and GCP GKE.





Block diagram for CCAT





Flow Chart for CCAT

**Work to Be Done**

1. Go to Amazon ECR Environment and List ECR Repositories

2. Remove Target Docker Image from ECR

3. Add a Backdoor Reverse Shell in the Pulled Docker Image

4. Check for Backdoored Docker Image

5. Push Backdoored Image Back to ECR

**Expected Outcomes**

1. A robust tool for **proactive vulnerability detection** in containers.
2. Improved security awareness through **real-world attack simulations**.
3. Multi-cloud compatibility, enabling **consistent security practices**.
4. Organizations and security teams will gain **deeper insights into container-specific threats** and best practices.

**References**

1. "Container Security in Cloud Environments: A Comprehensive Analysis and Future Directions for DevSecOps."
2. "Cybersecurity in Containerization Platforms: A Comparative Study of Security Challenges, Measures, and Best Practices."