Project Title: A medium interaction AI based adaptive honeypot mechanism

Name: Dayananda Bindhani

Enrolment No: 022200300004018

OVERVIEW

An AI based adaptive medium level interaction-based honeypot mechanism system which emulates the following services (ftp, ssh, smb, rdp, MySQL) to engage, deceive, and analyse cyber attacker's behaviours, their commands and method of attacks with extensive integration of AI/ML features.

SERVICE EMULATIONS

1. SSH:22
2. FTP:20
3. SMB:445
4. RDP:3389
5. MYSQL:3306

NOTE -: the reason behind choosing these services because, these services are most frequently attacked and are often choose as prime targets in the current scenario.

PRIMARY FEATURES

1. Service Emulation (ftp, ssh, smb, rdp, mysql)
2. Virtual Machine or containerization
3. AI features (dynamic responses, logging summery, attacks type detection forensic chain of custody generation)
4. Adaptive banner & response generation
5. Logging & forensic chain analysis emulator
6. Repayable logging functionality
7. Report generation

CORE-COMPONENTS

- AI-ENGINE
- SERVICE EMULATORS
- LOGGER
- CONTAINER
- VIRTUAL MACHINE
- VISUALIZATION & ANALYSYS

PRIMARY WORKING FLOW PROCESS

1. Spawn docker container and service emulator initialization
2. Create thread for each service emulators and listen on the redirection ports
3. if connection arrives, perform port redirection from the original service port to honeypot service port
4. dynamic banner generation and displaying to the intruder for specific service establishment
5. replay of fake response through dynamic content generation with help of AI engine models as well as pre-configured filesystem and other configurations.
6. Generate attackers command response through LLMs with correct context and realistic time delay, system load indicators and environ establishment.
7. Real time behavioural analysis and dynamic environ modification.
8. Generate multi-layer logging streams and store every different logging format in a secure virtual environment.
9. Generate repayable logging and forensic chain of attacks and store in their corresponding formats.
10. Creating summery of the entire logging simulation and produce visualization.
11. Close the connection by responding the corresponding service connection closing statement.

USPs

1. Dynamic policy & response changes without human intervention through it.
2. Forensic chain of attack visualization of all the attacker's behaviour and command injection to the specific service.

AI ACTIONS & FEATURES

1. Can generate dynamic real-time responses on according to commands.
2. Analysis of log files, credentials, command, to provide summery.
3. Can provide prevention techniques through attack types analysis.
4. Generate comprehensive reports.
5. Create forensic chain of attack visualization formats from attacker behaviour and command executions.

RECOMMENDED TECHNOLOGY ARCHITECTURE

Primary programming language: python

Service emulations references

- ssh (cowrie/qeeqbox/datatrap)
- ftp (qeeqbox/honeypot-ftp)
- smb (dionaea/qeeqbox)
- rdp (rdpy/qeebox)
- mysql (qeeqbox/datatrap)

ML libraries

- **scikit-learn**, **XGBoost**, **PyCaret** for fast anomaly detection, clustering, and ensemble models.
- **PyTorch** or **TensorFlow** for deep learning (for LLMs, CNN/LSTMs, or custom reinforcement learning logic).
- SentenceTransformers or spaCy for embedding and NLU tasks (useful for "cure the attacker" dynamics and realistic responses).

LLMs for Emulated Interaction

1. **Llama3 8B**: Strong balance of speed/quality for on-device response modeling in Linux shell/service mimicry.
2. For generative backends, **Ollama**, **vLLM**.

Adaptive logic and dynamic policy changes

1. FastAPI or Flask
2. Celery/RQ for queued tasks

Logging, Forensics, Reporting

1. Elastic search
2. PostgreSQL
3. Json
4. Web based interface (optional)

Visualization

1. Grafana
2. Kibana

Containerization

1. Docker/docker-compose (containered application for both windows and linux)
2. Kvm (vm specifically for linux aarch..)
3. Qemu (vm for linux aarch..)

UI/UX

Current interface: CLI (Command Line interface)

Future integration: GUI (graphical interface)

COMPOMENTS-ARCHITECTURE

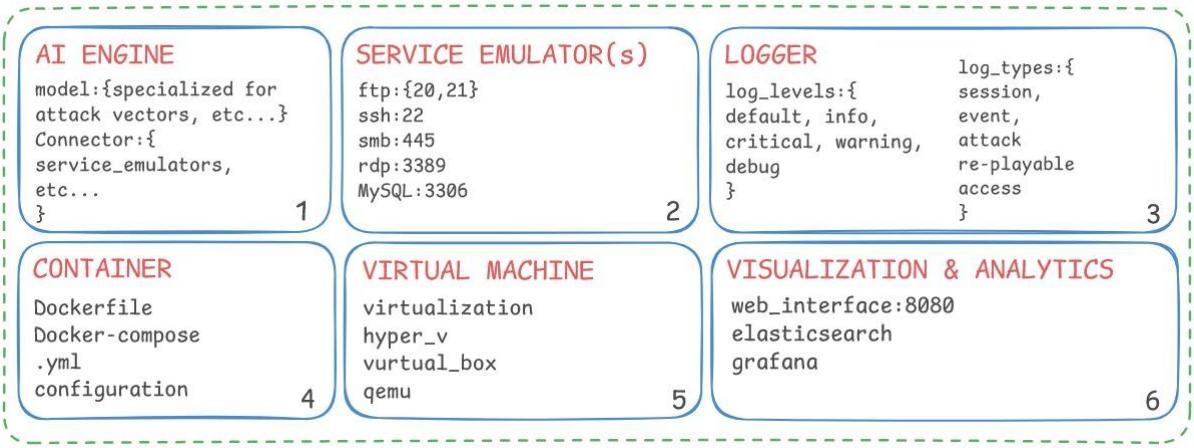

Fig.; 1

CORE-ARCHITECTURE

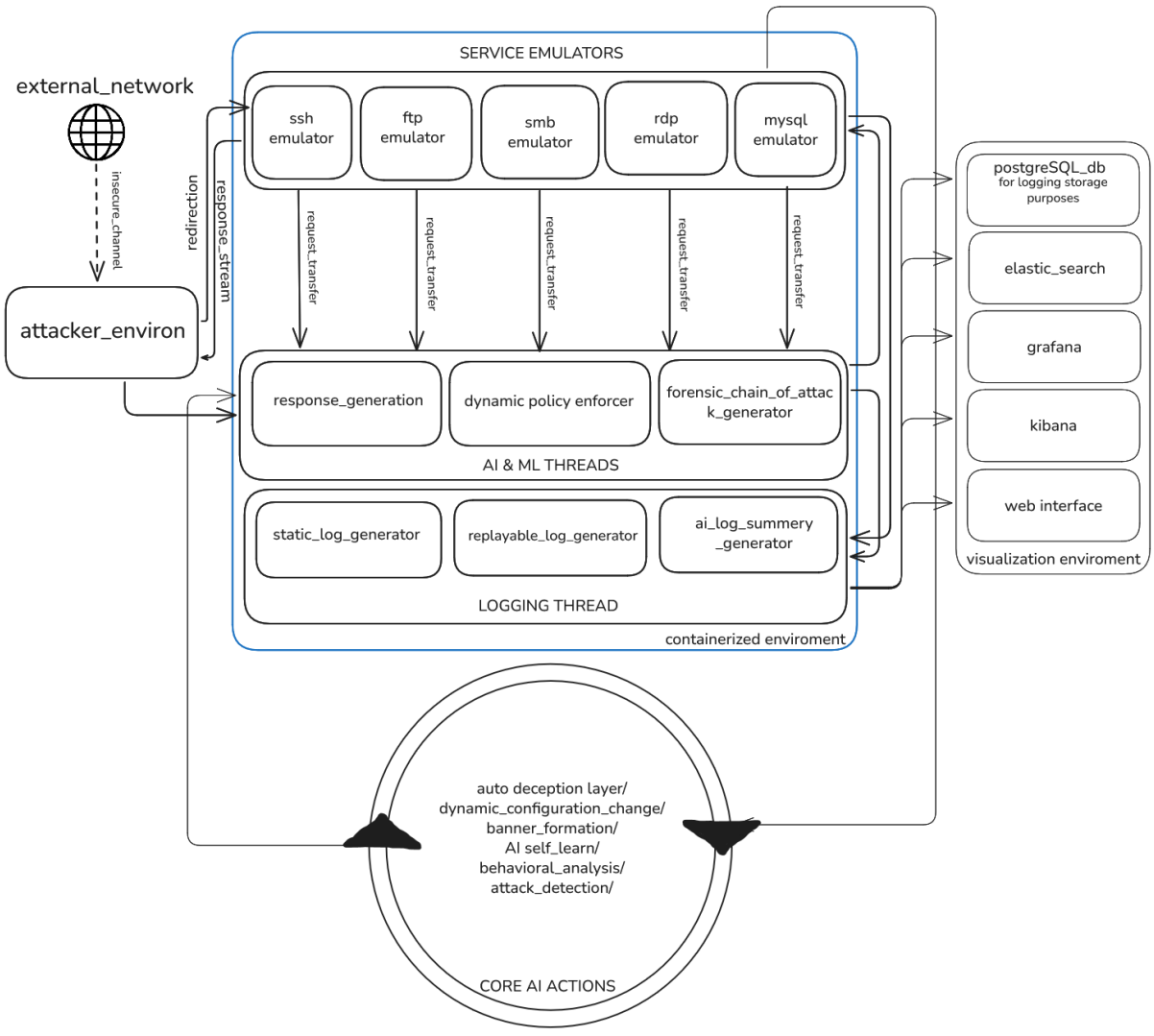

Fig.; 2