SentientRAT

SentientRAT is an open-source, modular multi-agent system designed for penetration testing, featuring a modern, stylish web-based GUI inspired by apps like ChatGPT and Instagram. It supports natural language interaction, multiple large language models (LLMs), dynamic knowledge ingestion, and interactive visualizations, making it ideal for both beginners and advanced cybersecurity professionals.

Table of Contents

Features

Modern GUI: Sleek React-based interface with dark/light mode, animations (Framer Motion), and a responsive design.
Natural Language Interaction: Chat with the agent using commands like "scan 192.168.1.5 with nmap -A" or "summarize pentest_guide.pdf".
LLM Support: Choose from MiniMax M1, DeepSeek R1, or LLaMA 3.1-8B, with GUI-based model downloads tailored to your system (GPU/CPU).
Dynamic Knowledge: Ingest PDFs/texts from knowledge/ for retrieval-augmented generation (RAG) queries.
Interactive Visualizations: Plotly charts for tool outputs (e.g., Nmap ports, sqlmap databases) with hover/zoom effects.
Pentesting Tools: Run Nmap, sqlmap, Hydra, and more in a Docker sandbox for secure execution.
Web Search: Combine DuckDuckGo/Brave APIs with LLM summarization for real-time intelligence.
Cloud-Ready: Deploy via Docker on AWS ECS, GCP Cloud Run, or Azure Container Instances.

Prerequisites

For Beginners

Operating System: Ubuntu 20.04+ (Linux), Windows 10/11 with WSL2, or macOS 12+.
Hardware:
CPU: 4+ cores, 16GB RAM (for LLaMA 3.1-8B).

GPU: Optional (NVIDIA RTX 3060+ for MiniMax/DeepSeek).
Disk: 50GB free (for models and Docker images).

Software:
Python 3.10
Docker Desktop (or Docker CLI on Linux)
Node.js 16+ and npm
Git

Internet: Required for downloading dependencies and models from Hugging Face.

For Advanced Users

Optional:
Brave API key for enhanced web search (get from Brave Search API).
NVIDIA drivers/CUDA for GPU-accelerated LLMs.
Cloud provider account (AWS/GCP/Azure) for deployment.

Customizations:
Custom model paths via MODEL_DIR environment variable.
Additional pentesting tools in tools_config.yaml.

Setup Instructions
Beginner Setup
This guide is designed for users new to cybersecurity or software setup. Each step includes explanations to help you understand the process.

Clone the Repository:

What: Download the SentientRAT source code from GitHub.
How: Open a terminal (on Windows, use PowerShell or WSL2) and run:git clone https://github.com/your-repo/SentientRAT.git
cd SentientRAT

Why: This creates a local copy of the project files.

Install System Dependencies:

What: Install Python, Docker, Node.js, and npm.
How (Ubuntu):sudo apt-get update
sudo apt-get install -y python3.10 python3-pip docker.io nodejs npm git
sudo systemctl start docker
sudo systemctl enable docker

For Windows/macOS, install Docker Desktop, Python 3.10, and Node.js.

Why: These tools are required to run the backend, GUI, and containerized tools.

Run Setup Script:

What: Install Python and Node.js dependencies and build the GUI.
How:bash scripts/setup.sh

Why: The script automates dependency installation, creates directories (knowledge/, memory/, models/), and builds the React app.
Output: You'll see "Setup complete" when finished.

Add Knowledge Files:

What: Place PDFs or text files for the agent to use (e.g., pentesting guides).
How: Copy files to the knowledge/ directory:cp ~/Downloads/pentest_guide.pdf knowledge/

Why: These files enable RAG queries (e.g., "summarize pentest_guide.pdf").

Verify Setup:

What: Check that directories and files are in place.
How: Run:ls -R

Expected: See main_agent.py, gui/, knowledge/, memory/, models/, etc.

Why: Ensures all files are correctly placed before running.

Advanced Setup
For experienced users who want to customize or optimize the setup.

Clone and Navigate:

```
git clone https://github.com/your-repo/SentientRAT.git
cd SentientRAT
```

Custom Dependencies:

Manually install specific versions or additional tools:pip install -r requirements.txt

```
cd gui
npm install
npm run build
```

Add tools to tools_config.yaml (e.g., Metasploit).

Environment Variables:

Create a .env file for custom settings:echo "LLM_MODEL=deepseek/r1" >> .env

```
echo "MODEL_DIR=/custom/path/models" >> .env
echo "BRAVE_API_KEY=your_key" >> .env
```

Load with export $(cat .env | xargs) or use in Docker.

GPU Optimization:

Install NVIDIA drivers and CUDA:sudo apt-get install -y nvidia-driver-535 nvidia-utils-535

Verify: nvidia-smi should show GPU info.

Custom Knowledge:

Add multiple PDFs/texts to knowledge/ or mount a custom directory in Docker.

Running the Application
Local Run

Start the Server:

What: Run the FastAPI backend and serve the GUI.
How:python main_agent.py

Output: See “Uvicorn running on http://0.0.0.0:8000”.
Why: This starts the agent and GUI.

Access GUI:

Open a browser and go to http://localhost:8000.
Use the chat interface to send commands or the Model Manager to download LLMs.

Test Commands:

Run the test script to verify functionality:python scripts/test_commands.py

Output: See responses for commands like “scan 192.168.1.5 with nmap”.

Docker Run

Build Docker Image:

What: Package the app into a container.
How:docker build -t sentientrat .

Why: Ensures consistent execution across environments.

Run Container:

What: Start the container with persistent storage.
How:docker run -d -p 8000:8000
-e LLM_MODEL=minimax/m1
-v $(pwd)/knowledge:/app/knowledge
-v $(pwd)/memory:/app/memory
-v $(pwd)/models:/app/models
sentientrat

Why: Maps local directories for knowledge, memory, and models.

Access GUI:

Open http://localhost:8000 in a browser.

Cloud Deployment

Push Image to Registry:

What: Upload the Docker image to a cloud registry (e.g., Docker Hub).
How:

```
docker tag sentientrat your-registry/sentientrat
docker push your-registry/sentientrat
```

Why: Makes the image available for cloud deployment.

Deploy to Cloud:

AWS ECS:
Create a task definition with the image your-registry/sentientrat.
Set environment variables (LLM_MODEL, BRAVE_API_KEY).
Expose port 8000.

GCP Cloud Run:

```
gcloud run deploy sentientrat
--image your-registry/sentientrat
--set-env-vars LLM_MODEL=minimax/m1
--port 8000
```

Azure Container Instances:
Use Azure CLI to deploy with similar settings.

Access GUI:

Use the cloud endpoint (e.g., https://your-service.a.run.app:8000).

Usage Examples

Chat Commands:
“Use nmap to scan 192.168.1.5 with aggressive mode and visualize” → Runs nmap -A 192.168.1.5 and shows a Plotly chart.
“Summarize all my pentesting books” → Summarizes PDFs in knowledge/.
“Search for recent CVEs for Apache” → Queries web APIs and returns a summary.
“Use hydra to brute force SSH on 192.168.1.5 with username admin” → Executes Hydra in a sandbox.

Model Downloads:
In the Model Manager tab, select “LLaMA 3.1-8B” for CPU or “MiniMax M1” for GPU, then click “Download Model”.

Troubleshooting

Error	Cause	Solution
ModuleNotFoundError	Missing Python dependency	Run pip install -r requirements.txt.

npm install fails
Node.js version mismatch
Install Node.js 16+: sudo apt-get install -y nodejs npm.

Docker build fails
Missing Docker daemon
Start Docker: sudo systemctl start docker.

Port 8000 in use
Another app using port
Change port: uvicorn main_agent:app --host 0.0.0.0 --port 8080.

Model download fails
No internet or Hugging Face rate limit
Check connectivity; retry later or use a Hugging Face token in .env.

GPU not detected
Missing NVIDIA drivers
Install drivers: sudo apt-get install -y nvidia-driver-535.

No response from GUI
Backend not running
Ensure python main_agent.py is active or container is running.

Permission denied
Insufficient permissions
Run with sudo or fix ownership: sudo chown -R $USER ..

General Tips:

Check logs in logs/sentientrat.log for detailed errors.
Restart Docker if containers fail: sudo systemctl restart docker.
Clear model cache if downloads fail: rm -rf models/*.

Contributing
We welcome contributions! To contribute:

Fork the repository on GitHub.
Create a branch: git checkout -b feature/your-feature.
Make changes and commit: git commit -m "Add your feature".
Push and create a pull request: git push origin feature/your-feature.
Add knowledge files to knowledge/ or enhance tools in tools_config.yaml.

Please follow the Code of Conduct and test changes before submitting.
License
SentientRAT is licensed under the MIT License and Apache-2.0 License, ensuring compliance with dependencies (React, Framer Motion, LangChain, MiniMax M1). See LICENSE for details.
Contact

Issues: Report bugs or request features on GitHub Issues.
Email: Contact the maintainers at support@your-repo.com.
Community: Join our Discord server (link TBD).

Thank you for using SentientRAT! Happy pentesting!