Bandit-NG

An interactive web-based terminal application for learning cybersecurity through the OverTheWire Bandit wargame, featuring an AI mentor system that provides guidance without giving away answers.

Features

- **Live SSH Terminal**: Direct connection to OverTheWire Bandit servers
- **AI Mentor**: Neo-themed assistant that explains concepts without spoiling solutions
- **Command Filtering**: Prevents cheating by redacting exact commands in AI responses
- **Rate Limited**: Responsible usage with 1 request per 90 seconds
- **Modern UI**: Matrix-themed interface with XTerm.js terminal emulation

Prerequisites

- Python 3.11+
- uv for package management
- Ollama running locally with Qwen2.5:1.5b model

Quick Start

1. **Clone the repository**

```bash
git clone <repository-url>
cd bandit-ng
```

2. **Set up virtual environment with uv**

```bash
uv venv
source .venv/bin/activate  # On Windows: .venv\Scripts\activate
```

3. **Install dependencies**

```bash
uv pip install -r requirements.txt
```

4. **Configure environment**

```bash
cp .env.example .env
# Edit .env with your settings if needed
```

5. **Set up Ollama**

```bash
# Install and start Ollama, then pull the required model
ollama pull qwen2.5:1.5b
```

6. **Run the application**

```bash
uvicorn app.main:app --reload --host 0.0.0.0 --port 8000
```

7. **Open your browser** Navigate to `http://localhost:8000`

Usage

1. The terminal on the right connects automatically to the Bandit servers
2. Work through the challenges as normal
3. When you need help, press `Ctrl+M` or click "Ask Mentor"
4. Neo will provide conceptual guidance without giving away exact commands

Project Structure

```
bandit-ng/
├── app/
│   ├── config.py      # Configuration management
│   ├── main.py        # FastAPI application
│   └── mentor.py      # AI mentor system
├── static/
│   ├── js/
│   │   ├── mentor.js  # Mentor interface
│   │   └── terminal.js # Terminal interface
│   ├── index.html     # Main application
│   └── styles.css     # Styling
├── .env.example       # Environment template
├── requirements.txt   # Python dependencies
└── README.md          # This file
```

Configuration

Environment variables (in `.env`):

- `OLLAMA_HOST`: Ollama API endpoint (default: http://localhost:11434)
- `GEMINI_API_KEY`: Gemini API key for future use (optional)

Development

Running in development mode

```bash
uvicorn app.main:app --reload
```

Code style

The project follows Python best practices with:

- Pydantic for configuration management
- FastAPI for the web framework
- Type hints throughout
- Environment-based configuration

Security & Educational Philosophy

This application is designed to help learning while preventing cheating:

- **Command Redaction**: AI responses filter out exact commands
- **Rate Limiting**: Prevents rapid-fire question abuse
- **Conceptual Focus**: Mentor explains concepts and links to documentation
- **No Answer Storage**: Doesn't persist or cache solutions

Contributing

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Update CHANGELOG.md
5. Submit a pull request

License

[Add your chosen license here]

Troubleshooting

Connection Issues

- Ensure Ollama is running and accessible
- Check that port 8000 is available
- Verify SSH access to bandit.labs.overthewire.org

Performance

- The Qwen2.5:1.5b model is optimized for speed over accuracy
- WebSocket connections may timeout; refresh if needed
- Rate limiting prevents too frequent mentor requests

Roadmap

- [ ] Add connection retry logic
- [ ] Implement user session management
- [ ] Add more AI model options
- [ ] Create progress tracking

- [ ] Add automated tests
- [ ] Implement logging system