Blockchain-Based Traffic Violation Logging & Penalty Management System

Richard Jong Wei Ze FYP 2024/2025-2 FCSIT | Network Computing

Table of Contents

1. System Requirements

Operating System: Ubuntu/Debian Linux

Architecture: ARM64 or x86_64

RAM: Minimum 4GB (8GB recommended)

Storage: Minimum 20GB free space

Network: Internet connection for downloads

2. Prerequisites Installation

1. System Updates and Basic Tools

```
sudo apt update && sudo apt upgrade -y
sudo apt install python3-pip git curl docker-compose jq -y
```

2. Docker Installation and Configuration

```
sudo systemctl start docker
sudo systemctl enable docker
sudo systemctl status docker
sudo usermod -a -G docker rj
sudo reboot
```

After reboot, verify Docker:

```
docker ps
```

3. Go Language Installation

```
cd ~/Downloads/
wget https://go.dev/dl/go1.24.2.linux-arm64.tar.gz
sudo tar -C /usr/local/ -xzf go1.24.2.linux-arm64.tar.gz
echo 'export PATH=$PATH:/usr/local/go/bin' >> ~/.profile
source ~/.profile
```

4. Node.js and npm Installation

```
sudo apt install nodejs npm -y
```

5. Verify All Installations

```
python3 --version
pip --version
docker --version
docker-compose --version
go version
curl --version
git --version
jq --version
node -v
npm -v
```

3. Hyperledger Fabric Network Setup

1. Create Project Structure

```
mkdir -p $HOME/go/src/github/<your_github_name>
cd $HOME/go/src/github/<your_github_name>
```

2. Install Hyperledger Fabric

```
curl -sSLO
https://raw.githubusercontent.com/hyperledger/fabric/main/scripts/insta
ll-fabric.sh
chmod +x install-fabric.sh
./install-fabric.sh
```

3. Install Docker Management (Portainer)

```
docker volume create portainer_data
docker run -d -p 8000:8000 -p 9000:9443 --name portainer --
restart=always \
  -v /var/run/docker.sock:/var/run/docker.sock \
  -v portainer_data:/data portainer/portainer-ce:lts
Access Portainer: https://<your_ip>:9000
Default credentials: Set during first login
```

4. Clone ChainGuard Repository

```
cd ~
git clone https://github.com/Kae-RZ/FYP_81049_ChainGuard.git
```

5. Configure Network Compose File

```
cd $HOME/go/src/github/<your_github_name>/fabric-samples/test-
network/compose
cp compose-test-net.yaml compose-test-net-bk.yaml
cp $HOME/FYP_81049_ChainGuard/fabric_network/compose-test-net.yaml ./
```

6. Setup Chaincode (Smart Contract)

```
cd $HOME/go/src/github/<your_github_name>/fabric-samples/
mkdir -p chaincode/trafficviolation
cd chaincode/trafficviolation
cp $HOME/FYP_81049_ChainGuard/fabric_network/trafficviolation.go ./
go mod init trafficviolation
go mod tidy
```

7. Create CouchDB Index

```
mkdir -p META-INF/statedb/couchdb/indexes
cd META-INF/statedb/couchdb/indexes
```

```
cat > indexDriverID.json << EOF
{
    "index": {
        "fields": ["driverID"]
    },
    "name": "driverID-index",
    "type": "json"
}
EOF
```

8. Start Fabric Network

```
cd $HOME/go/src/github/<your_github_name>/fabric-samples/test-network
./network.sh up createChannel -ca
```

9. Deploy Chaincode

```
./network.sh deployCC -ccn trafficviolation -
ccp ../chaincode/trafficviolation/ -ccl go
```

10. Configure Monitoring (Prometheus & Grafana)

```
cd $HOME/go/src/github/<your_github_name>/fabric-samples/test-
network/prometheus-grafana
```

For cAdvisor:

```
cadvisor:
  image: gcr.io/cadvisor/cadvisor:latest
  privileged: true
  container_name: cadvisor
  volumes:
    - /:/rootfs:ro
    - /var/run:/var/run:ro
    - /sys:/sys:ro
    - /var/lib/docker/:/var/lib/docker:ro
    - /sys/fs/cgroup:/sys/fs/cgroup:ro
  ports:
    - 8080:8080
  restart: always
```

For Grafana (change port to avoid conflict):

```
grafana:
  image: grafana/grafana:8.3.4
  container_name: grafana
  user: "104"
```

```
  depends_on:
    - prometheus
  ports:
    - 3002:3000
  volumes:
    - grafana_storage:/var/lib/grafana
    - ./grafana/provisioning/:/etc/grafana/provisioning/
  env_file:
    - ./grafana/config.monitoring
  restart: always
```

Start monitoring services:

```
docker-compose up -d
```

11. Setup Hyperledger Explorer

```
cd $HOME/go/src/github/<your_github_name>
mkdir explorer && cd explorer
wget https://raw.githubusercontent.com/hyperledger/blockchain-explorer/main/examples/net1/config.json
mkdir -p connection-profile
wget https://raw.githubusercontent.com/hyperledger/blockchain-explorer/main/examples/net1/connection-profile/test-network.json -P connection-profile
wget https://raw.githubusercontent.com/hyperledger/blockchain-explorer/main/docker-compose.yaml
sudo cp -r ../fabric-samples/test-network/organizations .
sudo chown -R $(id -u):$(id -g) organizations connection-profile/
```

Create environment file:

```
cat > .env << EOF
PORT=8081
EXPLORER_CONFIG_FILE_PATH=./config.json
EXPLORER_PROFILE_DIR_PATH=./connection-profile
FABRIC_CRYPTO_PATH=./organizations
EOF
```

Setup and start Explorer:

```
cp $HOME/FYP_81049_ChainGuard/setup_explorer.sh ./
chmod +x setup_explorer.sh
./setup_explorer.sh
```

4. Node.js API Server Configuration

1. Create API Directory

```
cd ~
mkdir fabric_sdk && cd fabric_sdk
cp $HOME/FYP_81049_ChainGuard/fabric_sdk/* ./
mkdir network/ wallet/
```

2. Copy Connection Profiles

```
cp $HOME/go/src/github/<your_github_name>/fabric-samples/test-network/organizations/peerOrganizations/org1.example.com/connection-org1.json network/
cp $HOME/go/src/github/<your_github_name>/fabric-samples/test-network/organizations/peerOrganizations/org2.example.com/connection-org2.json network/
```

3. Install Dependencies

```
npm init -y
npm install express fabric-network fabric-ca-client uuid
```

4. Test Connection

```
node testConnection.js
```

5. Manual Start

```
node app.js
```

6. Docker Approach (Recommended)

   Create .dockerignore:

```
cat > .dockerignore << EOF
node_modules
npm-debug.log
.git
.gitignore
EOF
```

   Create Dockerfile:

```
cat > Dockerfile << EOF
FROM node:18

WORKDIR /usr/src/app

COPY package*.json ./
```

```
RUN npm install

COPY . .

EXPOSE 3000
CMD [ "node", "app.js" ]
EOF
```

Build and run:

```
docker build -t fabric_sdk .
docker run --network="host" \
  -v $(pwd)/wallet:/usr/src/app/wallet \
  -v $(pwd)/network:/usr/src/app/network \
  --name fabricAPI fabric_sdk
```

5. Web Application Deployment

1. Install Web Server Components

```
sudo apt update
sudo apt install apache2 php libapache2-mod-php php-mongodb php-curl
php-mbstring -y
```

2. Setup Application Directory

```
cd /var/www/html
sudo mkdir chainguard
sudo chown $USER:$USER /var/www/html/chainguard
cp $HOME/FYP_81049_ChainGuard/web_ui/* /var/www/html/chainguard/
```

3. Install MongoDB

```
cd /var/www/html/chainguard
mkdir mongodb_data
docker run --name mongodb -d -p 27017:27017 --restart always \
  -v /var/www/html/chainguard/mongodb_data:/data/db mongo:latest
```

4. Install Composer

```
cd ~
curl -sS https://getcomposer.org/installer -o /tmp/composer-setup.php
HASH=`curl -sS https://composer.github.io/installer.sig`
php -r "if (hash_file('SHA384', '/tmp/composer-setup.php') === '$HASH')
{ echo 'Installer verified'; } else { echo 'Installer corrupt';
unlink('composer-setup.php'); } echo PHP_EOL;"
sudo php /tmp/composer-setup.php --install-dir=/usr/local/bin --
filename=composer
```

5. Install PHP Dependencies

```
cd /var/www/html/chainguard
composer require mongodb/mongodb stripe/stripe-php phpmailer/phpmailer
```

6. System Configuration

1. Apache Configuration

   Edit Apache site configuration:

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

   Add this configuration:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/chainguard

    <Directory /var/www/html/chainguard>
        Options -Indexes
        AllowOverride All
        Require all granted
    </Directory>
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

2. Setup .htaccess

```
cat > /var/www/html/chainguard/.htaccess << EOF
# Disable directory browsing
Options -Indexes
# Redirect root to the login page
DirectoryIndex /php/login.php
EOF
```

3. Enable URL Rewriting and Restart

```
sudo a2enmod rewrite
sudo chown -R www-data:www-data /var/www/html/chainguard
sudo chmod -R 755 /var/www/html/chainguard
sudo systemctl restart apache2
```

4. Configure Sensitive Information

   ⚠️ IMPORTANT: Update these files with your actual credentials:

```
payment_success.php (Line 18) - Stripe secret key
process_payment.php (Line 23) - Stripe secret key
pending_violation.js (Line 26) - Stripe public key
process_payment.php (Lines 41-42) - Update URLs with your IP address
send_email_notification.php (Lines 14-15) - Email credentials
```

7. Automation Scripts

1. Copy Automation Scripts

```
cp $HOME/FYP_81049_ChainGuard/setup_fabric.sh ~/
cp $HOME/FYP_81049_ChainGuard/shutdown_fabric.sh ~/
chmod +x ~/setup_fabric.sh ~/shutdown_fabric.sh
```

2. Usage

   Start the complete Fabric network:

```
./setup_fabric.sh
Shutdown the network:
./shutdown_fabric.sh
```

   Note: Logs are generated in ~/fabric_logs/

8. Service URLs

After successful installation, access these services:

Service	URL	Default Credentials
Main Website	http://<ip>/	Set during registration
Portainer	https://<ip>:9000/	Set during first login
Grafana	http://<ip>:3002/	admin/admin
CouchDB	http://<ip>:5984/_utils/	admin/adminpw
Hyperledger Explorer	http://<ip>:8081/	-
MongoDB Compass	mongodb://<ip>:27017	-

9. Troubleshooting

1. Network Issues

```
# Check if network is running
cd $HOME/go/src/github/<your_github_name>/fabric-samples/test-network
./network.sh down
./network.sh up createChannel -ca
```

2. Docker Issues

```
# Check running containers
docker ps -a

# Check container logs
docker logs <container_name>

# Restart specific service
docker restart <container_name>
```

3. MongoDB Issues

```
# Check MongoDB logs
docker logs mongodb

# Restart MongoDB
docker restart mongodb
```

4. API Connection Issues

```
# Check API logs
docker logs fabricAPI

# Test connection manually
node testConnection.js
```

5. Web Application Issues

```
# Check Apache logs
sudo tail -f /var/log/apache2/error.log

# Restart Apache
sudo systemctl restart apache2

# Check PHP configuration
php -m | grep mongodb
```

6. Test Environment Setup

   For external testing access:

```
sudo npm install -g localtunnel

# Expose different services
lt --port 80      # Web application
lt --port 9000    # Portainer
lt --port 27017   # MongoDB (WARNING: Others can access the database)
lt --port 3002    # Grafana
```

7. File Structure Overview

```
chainguard/
├── css/                    # Stylesheets
├── php/                    # PHP backend files
├── scripts/                # JavaScript files
├── mongodb_data/           # MongoDB data directory
├── .htaccess              # Apache configuration
└── composer.json          # PHP dependencies
```

8. Support

   For issues and support:

   a. Check the troubleshooting section above

   b. Review log files in ~/fabric_logs/

   c. Verify all services are running with docker ps

   d. Ensure all prerequisites are properly installed

Installation Complete!

ChainGuard system should now be fully operational with all components running and accessible via the provided URLs.