IDS Pipeline - Intrusion Detection System with ML Enhancement

A high-performance Intrusion Detection System combining Suricata, Kafka, and Machine Learning for real-time threat detection and analysis.

📚 Documentation

- 🏗️ **PIPELINE_ARCHITECTURE.md** - Detailed explanation of how both AF_PACKET and DPDK pipelines work
- 🚀 **NEXT_STEPS.md** - Roadmap and future development plans
- 📝 **CLEANUP_REPORT.md** - Recent codebase cleanup details

🚀 Quick Start

Choose Your Mode:

1. AF_PACKET Mode (Recommended for most users)

- Works with ANY network interface (including USB adapters)
- No special drivers required
- Easy setup

```
sudo ./run_afpacket_mode.sh
```

2. DPDK Mode (High-performance)

- Requires DPDK-compatible NIC
- Kernel bypass for maximum throughput
- More complex setup

PROF

```
sudo ./run_dpdk_mode.sh
```

🏷️ Architecture

```
Network Interface → Suricata IDS → Kafka → ML Consumer → Enhanced Alerts
                                    ↓
                          Signature Detection
                                    ↓
                          Flow Extraction
                                    ↓
                          ML Inference
```

```
                              ↓
                    Combined Scoring
```

🔧 Prerequisites

Required Software:

- **Suricata** (IDS engine)
- **Apache Kafka** (Message broker)
- **Python 3.8+** (ML processing)
- **tcpreplay** (Traffic replay - optional)

For AF_PACKET Mode:

- Any Linux network interface
- No special drivers needed

For DPDK Mode:

- DPDK-compatible NIC (Intel, Mellanox, etc.)
- Suricata compiled with DPDK support
- DPDK libraries installed
- Hugepages configured (2GB recommended)

📦 Installation

1. Install Dependencies

```
# Update system
sudo apt update && sudo apt upgrade -y

# Install Suricata
sudo apt install suricata -y

# Install Kafka (if not installed)
wget https://downloads.apache.org/kafka/3.6.0/kafka_2.13-3.6.0.tgz
tar -xzf kafka_2.13-3.6.0.tgz
sudo mv kafka_2.13-3.6.0 /usr/local/kafka

# Install Python dependencies
pip install -r requirements.txt

# Install tcpreplay (optional, for traffic replay)
sudo apt install tcpreplay -y
```

PROF

2. Configure the Pipeline

Edit the configuration file:

```
nano dpdk_suricata_ml_pipeline/config/pipeline.conf
```

Key settings:

- `NETWORK_INTERFACE` - Your network interface name
- `ML_MODEL_PATH` - Path to your ML model
- `KAFKA_BOOTSTRAP_SERVERS` - Kafka server address

3. Choose Your Mode

For AF_PACKET Mode:

```
sudo ./run_afpacket_mode.sh start
```

For DPDK Mode:

```
sudo ./run_dpdk_mode.sh start
```

📖 Usage Guide

AF_PACKET Mode

The AF_PACKET script provides an interactive menu:

```
sudo ./run_afpacket_mode.sh
```

PROF

Menu Options:

1. Start Complete Pipeline - Starts all components
2. Start Kafka Only
3. Start Suricata Only
4. Start ML Consumer Only
5. Start Kafka Bridge Only
6. Replay Traffic - Replay PCAP files
7. Check Status - View system status
8. View Logs - Monitor logs in real-time
9. Setup External Capture - Configure for external traffic
10. Stop All Services

Command-line Usage:

```
sudo ./run_afpacket_mode.sh start    # Start everything
sudo ./run_afpacket_mode.sh status   # Check status
sudo ./run_afpacket_mode.sh stop     # Stop all services
sudo ./run_afpacket_mode.sh logs     # View logs
```

DPDK Mode

The DPDK script includes additional DPDK-specific options:

```
sudo ./run_dpdk_mode.sh
```

Menu Options:

1. Start Complete Pipeline
2. Start Kafka Only
3. Start Suricata Only (DPDK)
4. Start ML Consumer Only
5. Start Kafka Bridge Only
6. Bind Interface to DPDK - Bind NIC to DPDK driver
7. Unbind Interface from DPDK - Restore NIC to kernel
8. Check Status
9. View Logs
10. Show DPDK Info - Display DPDK device status
11. Stop All Services

Command-line Usage:

```
sudo ./run_dpdk_mode.sh start     # Start everything
sudo ./run_dpdk_mode.sh bind      # Bind interface to DPDK
sudo ./run_dpdk_mode.sh unbind    # Unbind interface
sudo ./run_dpdk_mode.sh status    # Check status
sudo ./run_dpdk_mode.sh stop      # Stop all services
```

PROF

🔍 Components

1. Suricata IDS

- Monitors network traffic
- Applies signature-based detection rules
- Generates flow and alert data
- Outputs to EVE JSON format

2. Kafka Message Broker

- Receives Suricata events
- Provides reliable message queuing
- Enables distributed processing
- Topics: `suricata-alerts`, `ml-predictions`

3. Suricata-Kafka Bridge

- Reads Suricata EVE JSON logs
- Publishes events to Kafka
- Real-time streaming
- Auto-reconnection

4. ML Consumer

- Consumes events from Kafka
- Extracts CICIDS2017 features (65 features)
- Maps to model format (34 features)
- Performs ML inference
- Combines with Suricata alerts
- Publishes enhanced alerts

📊 Features

ML-Enhanced Detection

- **Flow-based analysis** - Every network flow analyzed by ML
- **65 CICIDS2017 features** - Industry-standard feature set
- **Multiple models** - Random Forest, LightGBM support
- **Real-time inference** - Low-latency predictions
- **Confidence scoring** - Probabilistic threat assessment

Performance

PROF

- **AF_PACKET Mode**: ~1-10 Gbps depending on hardware
- **DPDK Mode**: 10+ Gbps with kernel bypass
- **Scalable**: Kafka enables horizontal scaling
- **Efficient**: Batch processing for ML inference

📁 Project Structure

```
IDS/
├── run_afpacket_mode.sh          # AF_PACKET mode master script
├── run_dpdk_mode.sh              # DPDK mode master script
├── cleanup_codebase.sh           # Cleanup redundant files
├── requirements.txt              # Python dependencies
├── README.md                     # This file
│
├── config/
```

```
│   └── ids_config.yaml          # ML feature configuration
│
├── dpdk_suricata_ml_pipeline/
│   ├── config/
│   │   └── pipeline.conf        # Pipeline configuration
│   │
│   ├── scripts/                 # Individual component scripts
│   │   ├── 00_setup_external_capture.sh
│   │   ├── 01_bind_interface.sh
│   │   ├── 02_setup_kafka.sh
│   │   ├── 03_start_suricata.sh          (DPDK)
│   │   ├── 03_start_suricata_afpacket.sh (AF_PACKET)
│   │   ├── 04_start_ml_consumer.sh
│   │   ├── 05_replay_traffic.sh
│   │   ├── 06_start_kafka_bridge.sh
│   │   └── stop_all.sh
│   │
│   ├── src/                     # Python source code
│   │   ├── ml_kafka_consumer.py
│   │   ├── feature_extractor.py
│   │   ├── feature_mapper.py
│   │   ├── model_loader.py
│   │   └── alert_processor.py
│   │
│   ├── logs/                    # Log files
│   ├── models/                  # ML models
│   └── pcap_samples/           # Sample PCAP files
│
├── ML Models/                   # Pre-trained models
│   ├── random_forest_model_2017.joblib
│   └── lgb_model_2018.joblib
│
├── notebooks/                   # Jupyter notebooks
│   ├── CICIDS2017.ipynb
│   └── CICIDS2018.ipynb
│
└── tests/                      # Test scripts
    ├── quick_attack_demo.py
    └── quick_dpdk_test.py
```

PROF

🔧 Configuration

Pipeline Configuration (`dpdk_suricata_ml_pipeline/config/pipeline.conf`)

```
# Network Interface
NETWORK_INTERFACE="eth0"              # Your interface name
INTERFACE_PCI_ADDRESS=""              # Auto-detect (DPDK only)
DPDK_DRIVER="vfio-pci"               # DPDK driver

# DPDK Settings
```

```
DPDK_HUGEPAGES="2048"                # 2GB hugepages
DPDK_CORES="0,1"                     # CPU cores

# Suricata
SURICATA_CONFIG="/etc/suricata/suricata-dpdk.yaml"
SURICATA_CORES="2"                   # Worker threads
SURICATA_HOME_NET="192.168.0.0/16"   # Your network

# Kafka
KAFKA_BOOTSTRAP_SERVERS="localhost:9092"
KAFKA_TOPIC_ALERTS="suricata-alerts"
KAFKA_TOPIC_ML_PREDICTIONS="ml-predictions"

# ML Model
ML_MODEL_PATH="/path/to/model.joblib"
ML_CONFIDENCE_THRESHOLD="0.7"        # Alert threshold
```

📝 Logs

Log Locations:

- **Suricata**: `/var/log/suricata/suricata.log`
- **Suricata EVE JSON**: `/var/log/suricata/eve.json`
- **ML Consumer**: `dpdk_suricata_ml_pipeline/logs/ml/ml_consumer.log`
- **Kafka Bridge**: `dpdk_suricata_ml_pipeline/logs/kafka_bridge.log`

View Logs:

```
# Via menu
sudo ./run_afpacket_mode.sh
# Select option 8

# Direct access
tail -f /var/log/suricata/eve.json
tail -f dpdk_suricata_ml_pipeline/logs/ml/ml_consumer.log
```

PROF

🧪 Testing

Replay Sample Traffic:

```
sudo ./run_afpacket_mode.sh
# Select option 6 (Replay Traffic)

# Or directly:
sudo tcpreplay -i eth0 -M 10 pcap_samples/sample.pcap
```

Quick Attack Demo:

```
cd tests
python3 quick_attack_demo.py
```

🐛 Troubleshooting

Suricata Won't Start

```
# Check Suricata config
sudo suricata -T -c /etc/suricata/suricata.yaml

# Check interface is up
ip link show eth0
```

Kafka Connection Issues

```
# Check Kafka is running
pgrep -f kafka

# Test Kafka connectivity
kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic
suricata-alerts
```

DPDK Binding Issues

```
# Check DPDK devices
sudo dpdk-devbind.py --status

# Unbind and retry
sudo ./run_dpdk_mode.sh unbind
sudo ./run_dpdk_mode.sh bind
```

PROF

ML Consumer Errors

```
# Check Python dependencies
pip install -r requirements.txt

# Verify model path
ls -la /path/to/model.joblib
```

```
# Check logs
tail -f dpdk_suricata_ml_pipeline/logs/ml/ml_consumer.log
```

🧹 Cleanup

To remove redundant files and documentation:

```
sudo ./cleanup_codebase.sh
```

This will:

- Remove all duplicate PDF files
- Remove redundant documentation (30+ files)
- Remove legacy code directory
- Backup everything before deletion

📚 Additional Documentation

- **QUICKSTART.md** - Quick start guide
- **SETUP_GUIDE.md** - Detailed setup instructions
- **PRODUCTION_DPDK_GUIDE.md** - DPDK production deployment
- **REALTIME_PIPELINE_GUIDE.md** - Real-time processing guide
- **EXTERNAL_TRAFFIC_GUIDE.md** - External traffic capture setup
- **USB_ADAPTER_GUIDE.md** - USB network adapter usage
- **REMOTE_DEVICE_SETUP.md** - Remote device configuration

🤝 Contributing

Contributions are welcome! Please follow these steps:

PROF

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Test thoroughly
5. Submit a pull request

📄 License

This project is licensed under the MIT License.

👥 Authors

- **Sujay** - Initial work

🙏 Acknowledgments

- Suricata IDS team
- Apache Kafka project
- CICIDS2017 dataset creators
- DPDK community

📞 Support

For issues and questions:

- Check the troubleshooting section
- Review log files
- Open an issue on GitHub

Last Updated: October 2025