# Complete Pipeline Setup Guide

## Architecture

```
External Traffic (tcpreplay)
    ↓
Physical NIC (bound to DPDK)
    ↓
DPDK (zero-copy capture)
    ↓
Suricata IDS (DPDK mode)
    ↓
Kafka (eve-kafka output)
    ↓
ML Inference Engine
    ↓
Predictions & Alerts
```

## Prerequisites

Before starting, ensure you have:

- ✓ DPDK installed
- ✓ Suricata with DPDK support
- ✓ Python virtual environment with required packages
- ✓ At least 2GB of available RAM for hugepages
- ✓ A network interface you can take offline for DPDK binding

## Step-by-Step Setup

PROF

### Step 1: Install DPDK and Suricata

```
cd /home/sujay/Programming/IDS
sudo ./install_dpdk_suricata.sh
```

This installs:

- DPDK 23.11
- Suricata 7.0.7 with DPDK support
- Configures hugepages
- Sets up system dependencies

**Time:** ~20-30 minutes

### Step 2: Reboot (Recommended)

```
sudo reboot
```

This ensures hugepages are properly allocated.

### Step 3: Configure Pipeline

Edit the configuration file:

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

Key settings to configure:

- `NETWORK_INTERFACE="eth0"` - Interface to bind to DPDK
- `SURICATA_HOME_NET="192.168.0.0/16"` - Your network range
- `ML_MODEL_PATH="../ML Models/random_forest_model_2017.joblib"` - ML model

### Step 4: Start Pipeline Components

**4a. Setup and Start Kafka**

```
cd dpdk_suricata_ml_pipeline/scripts
./02_setup_kafka.sh
```

This will:

PROF

- Download and install Kafka (if not present)
- Start Zookeeper and Kafka broker
- Create required topics
- Install Python Kafka libraries

**4b. Bind Network Interface to DPDK**

⚠ **WARNING:** This will take your network interface offline!

```
sudo ./01_bind_interface.sh
```

This will:

- Backup interface configuration

- Take interface down
- Load DPDK kernel modules
- Bind interface to DPDK driver (vfio-pci)

**Important:** If this is your primary network interface, you will lose network connectivity!

#### 4c. Start Suricata in DPDK Mode

```
sudo ./03_start_suricata.sh
```

This will:

- Generate Suricata configuration with DPDK and Kafka support
- Start Suricata to capture from DPDK interface
- Output alerts to Kafka topic

#### 4d. Start ML Inference Consumer

```
./04_start_ml_consumer.sh
```

This will:

- Activate Python virtual environment
- Start consumer reading from Kafka
- Perform ML inference on alerts
- Publish predictions to Kafka

### Step 5: Send Traffic

PROF

You have three options:

#### Option A: Replay PCAP File

```
sudo ./05_replay_traffic.sh ../pcap_samples/sample.pcap
```

Or with options:

```
sudo ./05_replay_traffic.sh capture.pcap -s 100 -l 5
```

#### Option B: External System

From another machine on the same network:

```
# Send test traffic
ping <target_ip>
curl http://<target_ip>
# Or use hping3, nmap, etc.
```

**Option C: Generate Synthetic Traffic**

Use scapy or other tools:

```
python3 << EOF
from scapy.all import *
send(IP(dst="192.168.1.100")/TCP(dport=80), count=1000)
EOF
```

### Step 6: Monitor Pipeline

Check pipeline status:

```
./status_check.sh
```

Monitor individual components:

**Suricata Logs:**

```
tail -f /var/log/suricata/eve.json
tail -f /var/log/suricata/suricata.log
```

PROF

**Kafka Alerts:**

```
kafka-console-consumer.sh --bootstrap-server localhost:9092 \
    --topic suricata-alerts --from-beginning
```

**ML Predictions:**

```
tail -f ../logs/ml/ml_consumer.log

# Or from Kafka:
kafka-console-consumer.sh --bootstrap-server localhost:9092 \
    --topic ml-predictions --from-beginning
```

### Step 7: Stop Pipeline

When finished:

```
sudo ./stop_all.sh
```

This will:

- Stop ML consumer
- Stop Suricata
- Optionally stop Kafka
- Optionally unbind DPDK interfaces

## Detailed Component Configuration

### DPDK Configuration

The DPDK binding script handles:

- **Interface Binding:** Binds your NIC to DPDK driver
- **Driver Options:** vfio-pci (default), uio_pci_generic, igb_uio
- **Hugepages:** 2GB allocated by default
- **Backup:** Saves original interface configuration

To manually manage:

```
# Check status
dpdk-devbind.py --status

# Bind interface
sudo dpdk-devbind.py -b vfio-pci 0000:02:00.0

# Unbind
sudo dpdk-devbind.py -u 0000:02:00.0

# Restore to original driver
sudo dpdk-devbind.py -b e1000e 0000:02:00.0
```

PROF

### Suricata Configuration

Key configuration in `/etc/suricata/suricata-dpdk.yaml`:

**DPDK Settings:**

```
dpdk:
  eal-params:
```

```
    proc-type: primary
  interfaces:
    - interface: 0000:02:00.0
      threads: 2
      cluster-id: 99
```

**Kafka Output:**

```
outputs:
  - eve-log:
      enabled: yes
      filetype: kafka
      kafka:
        bootstrap-servers: localhost:9092
        topic: suricata-alerts
        compression-codec: snappy
```

## Kafka Topics

Two topics are created:

1. **suricata-alerts:** Raw alerts from Suricata
    - 3 partitions
    - Snappy compression
    - JSON format
2. **ml-predictions:** ML inference results
    - 3 partitions
    - Includes original alert + prediction
    - Enhanced with confidence scores

PROF

## ML Model Requirements

The ML model should be:

- Scikit-learn compatible (joblib format)
- Trained on network flow features
- Binary classification (benign/attack)

Expected features:

- src_port, dest_port, protocol
- flow_duration, flow_pkts_toserver, flow_pkts_toclient
- flow_bytes_toserver, flow_bytes_toclient
- Computed: packet_rate, byte_rate, bytes_per_packet

# Troubleshooting

## Issue: Interface Binding Fails

**Error:** "Cannot bind device to vfio-pci"

**Solution:**

```
# Load vfio module
sudo modprobe vfio-pci

# Enable NOIOMMU mode if needed
echo 1 | sudo tee /sys/module/vfio/parameters/enable_unsafe_noiommu_mode

# Try binding again
sudo ./01_bind_interface.sh
```

## Issue: Suricata Won't Start

**Error:** "Failed to initialize DPDK"

**Check:**

1. Hugepages allocated: `grep Huge /proc/meminfo`
2. Interface bound: `dpdk-devbind.py --status`
3. DPDK support: `suricata --build-info | grep DPDK`

**Solution:**

```
# Allocate hugepages
echo 2 | sudo tee /sys/kernel/mm/hugepages/hugepages-
1048576kB/nr_hugepages

# Or reboot for GRUB settings to take effect
sudo reboot
```

PROF

## Issue: No Alerts in Kafka

**Check:**

1. Suricata running: `pgrep -x suricata`
2. Kafka running: `netstat -tuln | grep 9092`
3. Traffic flowing: `suricatasc -c dump-counters`

**Test Kafka manually:**

```
```

```
# Produce test message
echo '{"test": "message"}' | kafka-console-producer.sh \
    --bootstrap-server localhost:9092 --topic suricata-alerts

# Consume it
kafka-console-consumer.sh --bootstrap-server localhost:9092 \
    --topic suricata-alerts --from-beginning
```

### Issue: ML Consumer Not Processing

**Check logs:**

```
tail -f ../logs/ml/ml_consumer.log
tail -f ../logs/ml/ml_consumer.out
```

**Common issues:**

- Model file not found
- Kafka connection failed
- Feature extraction errors

**Solution:**

```
# Restart consumer with debug logging
pkill -f ml_kafka_consumer.py
python3 ../src/ml_kafka_consumer.py --config ../config/pipeline.conf
```

### Issue: tcpreplay Not Working

PROF

**Error:** "Fatal Error: Can't send packet"

This usually means interface is bound to DPDK. Options:

1. Use another interface for replay
2. Unbind interface temporarily
3. Send traffic from external system

### Performance Issues

**Symptom:** Packet drops, high CPU usage

**Optimize:**

1. **Increase Hugepages:**

```
```

```
echo 4 | sudo tee /sys/kernel/mm/hugepages/hugepages-
1048576kB/nr_hugepages
```

2. **CPU Isolation:**
   Add to `/etc/default/grub`:

   ```
   GRUB_CMDLINE_LINUX="isolcpus=1,2,3 nohz_full=1,2,3"
   ```

3. **Increase Suricata Workers:**
   Edit `config/pipeline.conf`:

   ```
   SURICATA_CORES="4"
   ```

4. **Tune Suricata Rules:**
   Disable unused rules to reduce processing load

# Advanced Usage

## Custom ML Models

To use your own model:

1. Train model on appropriate features
2. Save as joblib: `joblib.dump(model, 'my_model.joblib')`
3. Update config: `ML_MODEL_PATH="/path/to/my_model.joblib"`
4. Restart ML consumer

## Multiple Interfaces

PROF

To monitor multiple interfaces:

1. Bind multiple interfaces to DPDK
2. Configure Suricata with multiple DPDK interfaces
3. Run separate Suricata instances if needed

## Cluster Deployment

For production:

1. **Kafka Cluster:** Multiple brokers, replication
2. **Suricata Cluster:** Multiple workers, load balancing
3. **ML Cluster:** Multiple consumers, different models
4. **Storage:** ElasticSearch, PostgreSQL for alerts

## Integration with SIEM

Forward predictions to SIEM:

```
# Add to ml_kafka_consumer.py
def send_to_siem(prediction):
    if prediction['confidence'] > 0.9:
        # Send to Splunk, ELK, etc.
        pass
```

## Performance Benchmarks

Expected performance (single interface, 4-core system):

- **DPDK:** 10+ Gbps packet capture
- **Suricata:** 1-5 Gbps with full rule set
- **Kafka:** 100K+ messages/sec
- **ML Inference:** 10K+ predictions/sec

## Useful Commands Reference

```
# Pipeline management
./status_check.sh              # Check all components
sudo ./stop_all.sh             # Stop everything
./02_setup_kafka.sh            # Restart Kafka

# DPDK
dpdk-devbind.py --status       # Show device bindings
dpdk-status                    # Full DPDK status

# Suricata
suricatasc -c stats            # Live statistics
suricatasc -c reload-rules     # Reload rules
suricata-update                # Update rules

# Kafka
kafka-topics.sh --list --bootstrap-server localhost:9092
kafka-consumer-groups.sh --bootstrap-server localhost:9092 --list
kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic
suricata-alerts

# Monitoring
htop                           # System resources
iftop -i eth0                  # Network traffic
iostat -x 1                    # Disk I/O
```

PROF

## Support and Resources

- **Pipeline Issues:** Check logs in `dpdk_suricata_ml_pipeline/logs/`

- **DPDK Documentation:** https://doc.dpdk.org/
- **Suricata Documentation:** https://docs.suricata.io/
- **Kafka Documentation:** https://kafka.apache.org/documentation/

---

**Last Updated:** October 2, 2025
**Version:** 1.0