Demo Day Guide - Step by Step

Pre-Demo Checklist (Do This Before Friday)

1. Test Everything

```
# Test 1: Enhanced detection system
python enhanced_main.py
# Let it run for 30 seconds, then Ctrl+C

# Test 2: Alert system
python alert_system.py
# Should show 4 colored alerts

# Test 3: Attack detection
# Terminal 1: python enhanced_main.py
# Terminal 2: python attack_simulator.py
# Watch for HIGH/MEDIUM alerts
```

2. Prepare Your Laptop

- Charge laptop fully
- Close unnecessary applications
- Open 2 terminal windows side by side
- Have PowerPoint presentation ready
- Test AWS credentials are working
- Clear old log files (optional):

```
del threat_alerts.log
del threat_alerts.json
```

3. Have These Files Open in Editor

- `enhanced_main.py` - to show code
- `alert_system.py` - to explain alerts
- `ids_engine.py` - to show IDS logic
- `ueba_engine.py` - to show UEBA logic
- `threat_fusion_engine.py` - to show fusion

Demo Script (15-20 minutes)

Part 1: Introduction (2 minutes)

What to Say: > "Good morning/afternoon. Today I'm presenting my Hybrid Threat Detection System that combines network intrusion detection with user behavior analytics for real-time security monitoring on AWS."

Show PowerPoint Slide 1: - Project title - Your name - Brief overview

Key Points: - Combines TWO detection methods (network + user behavior) - Real-time monitoring (10-second cycles) - AWS-native implementation - Production-ready with alerts and logging

---

Part 2: Problem Statement (2 minutes)

Show PowerPoint Slides 2-3: - Problem definition - Literature gap

What to Say: > "Traditional security systems monitor either network traffic OR user behavior separately. This creates blind spots. My system is the first to combine AWS CloudWatch network metrics with CloudTrail user behavior in real-time using weighted fusion."

Key Points: - Existing research: siloed approaches - Your novelty: hybrid real-time fusion - AWS-native implementation - Faster detection (10-20s vs 30-60s in literature)

Part 3: System Architecture (3 minutes)

Show PowerPoint Slide: - Architecture diagram

What to Say: > "The system has three main components: IDS Engine monitors network traffic from CloudWatch, UEBA Engine analyzes user behavior from CloudTrail, and Threat Fusion Engine combines both using 60/40 weighted scoring."

Open Code - Show `threat_fusion_engine.py`:

```python
def combine_risks(network_risk, user_risk):
    final_risk = (0.6 * network_risk) + (0.4 * user_risk)

    if final_risk > 0.8:
        level = "CRITICAL"
    elif final_risk > 0.6:
        level = "HIGH"
    # ...
```

Explain: - Network risk weighted 60% (immediate impact) - User risk weighted 40% (context) - Four threat levels: CRITICAL, HIGH, MEDIUM, LOW

Part 4: Live Demo - Normal Operation (3 minutes)

Terminal 1 - Start System:

```
python enhanced_main.py
```

What to Say: > "Let me show you the system in normal operation. I'll start the enhanced detection system."

Point Out on Screen: 1. System initialization message 2. Detection cycle starts 3. Network metrics: ~1,000 bytes, ~10-20 packets 4. Risk scores: Network 0.05, User 0.10, Final 0.07 5. Threat Level: LOW

What to Say: > "As you can see, in normal operation, we have minimal network traffic, low risk scores, and the system correctly classifies this as LOW threat. The system checks every 10 seconds."

Let it run for 2-3 cycles to show consistency

Part 5: Live Demo - Attack Detection (5 minutes)

Keep Terminal 1 running

Terminal 2 - Launch Attack:

```
python attack_simulator.py
```

What to Say: > "Now I'll simulate a DDoS attack using 300 concurrent threads sending HTTP requests to our EC2 instance. Watch what happens…"

Watch Terminal 1 - Point Out:

Within 10-20 seconds, you'll see: 1. Network traffic SPIKE: - Bytes: 1,000 → 1,000,000+ (1000x increase) - Packets: 10 → 10,000+ (1000x increase)

2. Risk scores INCREASE:
   - Network Risk: 0.05 → 0.85-0.95
   - User Risk: stays ~0.10 (normal)
   - Final Risk: 0.07 → 0.55-0.65
3. Threat Level ESCALATES:
   - LOW → MEDIUM or HIGH
4. **ALERT TRIGGERED:**
   - Color-coded console alert appears
   - Shows detailed threat information

What to Say: > "Within 15 seconds, the system detected the attack! Notice: > - Network traffic increased 1000x > - Network risk jumped to 0.95 > - But user behavior stayed normal at 0.10 > - This confirms it's an external attack, not insider threat > - The hybrid fusion gave us 0.61 final risk - HIGH threat level > - An alert was automatically triggered"

Wait for attack to complete (60 seconds)

What to Say: > "The attack simulator runs for 60 seconds. Notice the system continues to detect the elevated threat throughout the attack period."

Part 6: Alert System Demo (2 minutes)

Stop Terminal 1 (Ctrl+C)

Show Statistics: Point to the statistics output:

```
  SYSTEM STATISTICS
Uptime: 0:02:30
Detection Cycles: 15
Threats Detected: 5
Total Alerts: 5
   - CRITICAL: 0
   - HIGH: 3
   - MEDIUM: 2
   - LOW: 0
```

What to Say: > "The system tracked all detections and generated alerts. Let me show you the alert system features."

Terminal 2 - Test Alert System:

```
python alert_system.py
```

Point Out: 1. Color-coded alerts (LOW=green, MEDIUM=yellow, HIGH=orange, CRITICAL=red) 2. Detailed threat information 3. Alert statistics at the end

Show Log Files:

```
type threat_alerts.log
```

What to Say: > "All alerts are logged to files for analysis. We have both human-readable logs and JSON format for machine processing."

Part 7: Enhanced Features (2 minutes)

Show in Editor - `alert_system.py`:

Point Out: 1. **Multi-channel alerts:** - Console (color-coded) - Email (HTML formatted) - File logging (JSON + text)

2. **Rate limiting:**

   ```
   "max_emails_per_hour": 10
   ```

   - Prevents alert spam

3. **Configurable thresholds:**

   ```
   {
     "critical": 0.8,
     "high": 0.6,
     "medium": 0.4
   }
   ```

What to Say: > "The system includes production-ready features like rate limiting to prevent alert fatigue, configurable thresholds, and multiple notification channels including email alerts for HIGH and CRITICAL threats."

Part 8: Novelty & Contributions (2 minutes)

Show PowerPoint Slides: - Literature comparison - Novelty analysis

What to Say: > "Compared to existing research, my work has five key novelties:

1. **First AWS-native hybrid system** - No other research combines CloudWatch + CloudTrail in real-time
2. **Novel weighted fusion** - 60/40 weighting based on threat impact analysis
3. **Faster detection** - 10-20 seconds vs 30-60 seconds in literature
4. **Real attack validation** - Tested with actual DDoS attack, not just synthetic datasets
5. **Production-ready** - Complete with alerts, logging, and monitoring"

Show Performance Comparison Slide:

```
Detection Time:      10-20s   vs   30-60s (literature)    2-3x faster
False Positives:     0%       vs   15-20% (literature)    Better
Real Attack Test:    Yes      vs   Synthetic datasets     Validated
```

Part 9: Results & Performance (1 minute)

Show PowerPoint Slide: - Performance metrics table

What to Say: > "In testing, the system achieved: > - 1,242x traffic increase detected > - 10-20 second detection latency > - 0% false positives > - 100% true positive rate for DDoS attacks > - Minimal system overhead (<5% CPU)"

Part 10: Conclusion & Q&A (2 minutes)

Show Final PowerPoint Slides: - Summary - Future work

What to Say: > "To summarize, I've built the first AWS-native hybrid threat detection system that combines network and user behavior analytics in real-time. The system successfully detected a 1,000x traffic increase within 20 seconds with zero false positives, outperforming literature benchmarks."

Future Work: - Multi-region monitoring - Dashboard visualization (show dashboard.py code briefly) - Integration with additional AWS security services - Adaptive threshold learning

Open for Questions

Handling Common Questions

Q: "Why 60/40 weighting?"

A: "Network threats like DDoS have immediate impact, so they get higher weight (60%). User behavior provides context but manifests slower, so 40%. This weighting was validated through testing and aligns with threat landscape analysis from literature."

Q: "What about false positives?"

A: "The hybrid approach reduces false positives. If network risk is high but user behavior is normal, we know it's an external attack, not a false alarm. In testing, we achieved 0% false positives."

Q: "How does it scale?"

A: "AWS CloudWatch and CloudTrail scale automatically. Current bottleneck is 10-second polling. For production, we could move to streaming with CloudWatch Events and Lambda for sub-second detection."

Q: "What's the cost?"

A: "Minimal. CloudWatch API calls are ~$0.01 per 1000 requests. CloudTrail logs are already collected. Total cost: <$5/month for a single instance."

Q: "Can it detect other attacks?"

A: "Currently optimized for DDoS. The architecture is extensible - we can add detection for SQL injection, brute force, data exfiltration by adding more features to the UEBA engine."

Q: "Why not use AWS GuardDuty?"

A: "GuardDuty is excellent but expensive ($4.50/million events) and a black box. My system is transparent, customizable, and demonstrates the research contribution of hybrid fusion."

Q: "What about the dashboard?"

A: "I've built a web-based dashboard with real-time charts and monitoring. Due to time constraints in the demo, I'm showing the core detection system, but the dashboard code is complete and functional." (Show dashboard.py briefly if time permits)

Demo Day Checklist

Before Demo:

- ☐ Laptop charged
- ☐ AWS credentials working
- ☐ PowerPoint open
- ☐ 2 terminals ready
- ☐ Code editor open with key files
- ☐ Internet connection stable
- ☐ Clear old log files (optional)

During Demo:

- ☐ Speak clearly and confidently
- ☐ Make eye contact with reviewers
- ☐ Point to screen when explaining
- ☐ Don't rush - take your time
- ☐ If something fails, stay calm and explain

After Demo:

- ☐ Answer questions confidently
- ☐ Thank the reviewers
- ☐ Offer to show additional features if time permits

Timing Breakdown

Section	Time	What to Show
Introduction	2 min	PowerPoint intro
Problem Statement	2 min	PowerPoint slides
Architecture	3 min	PowerPoint + code
Normal Operation	3 min	Live demo - Terminal 1
Attack Detection	5 min	Live demo - Both terminals
Alert System	2 min	Alert demo + logs
Enhanced Features	2 min	Code walkthrough
Novelty	2 min	PowerPoint comparison
Results	1 min	PowerPoint metrics
Conclusion	2 min	PowerPoint + Q&A
TOTAL	**24 min**	**Includes buffer**

Pro Tips

1. Practice Run

Do a complete practice run the night before: - Time yourself - Practice what you'll say - Test all commands - Ensure AWS is working

2. Backup Plan

If live demo fails: - Show screenshots from PROJECT_OUTPUT_SUMMARY.md - Walk through code instead - Explain what should happen

3. Confidence Boosters

- You built this - you know it best
- The system works - you've tested it
- Your research is novel - you have proof
- Stay calm and explain clearly

4. Body Language

- Stand/sit up straight
- Make eye contact
- Use hand gestures to point at screen
- Smile and show enthusiasm

5. Voice

- Speak clearly and not too fast

- Pause between sections
- Emphasize key points
- Vary your tone to maintain interest

Final Checklist - Night Before

☐ Practice complete demo 2-3 times
☐ Test all commands work
☐ Charge laptop fully
☐ Prepare backup screenshots
☐ Review common questions
☐ Get good sleep
☐ Arrive early on demo day

Key Messages to Emphasize

1. **"First AWS-native hybrid system"** - Emphasize novelty
2. **"10-20 second detection"** - Faster than literature
3. **"0% false positives"** - Better accuracy
4. **"Real attack validation"** - Not just theory
5. **"Production-ready"** - Complete system

Emergency Contacts

If something goes wrong: - AWS not working? → Show code and explain what should happen - Attack simulator fails? → Use previous test results from PROJECT_OUTPUT_SUMMARY.md - Laptop crashes? → Have PowerPoint on USB backup

---

You're Ready!

Remember: - You've built an impressive system - Your research is novel and validated - The demo works - you've tested it - Stay confident and explain clearly - You've got this!

Good luck on Friday!