Implementation and Performance Analysis of Machine Learning-Based Intrusion Detection for DNP3 Protocol in SCADA Systems

Presented By
Team 2
V S M Bharat Kumar (2024202030)
Sai Charan Thammi (2024202021)
Aditya Sangana (2024202027)

International Institute of Information Technology, Hyderabad
Research in Information Security
Monsoon Semester 2025

Outline

Background

- **SCADA Systems:** Critical infrastructure control systems used in power generation, water treatment, oil & gas, transportation
- **DNP3 Protocol:** Distributed Network Protocol 3 - widely adopted communication protocol for industrial control systems (designed 1993)
- **Security Challenge:** Originally air-gapped systems now connected to IoT, TCP/IP networks
- **Vulnerability:** DNP3 lacks built-in authentication and encryption mechanisms
- **Consequences:** Attacks can lead to economic losses, service disruptions, safety threats

Problem Statement

Key Issues

- Traditional signature-based detection inadequate for DNP3 protocol vulnerabilities
- Need for intelligent, adaptive detection mechanisms
- Real-world attacks are rare - training on imbalanced data is critical
- Requirement for multi-class detection (identifying specific attack types)

Our Objective

Replicate and analyze a machine learning-based intrusion detection scheme for DNP3 with realistic intrusion rates (5%, 10%, 15%)

Network Model

Components:

- Master Station (Control Unit)
- Remote Terminal Units (RTUs)
- Communication Infrastructure (TCP/IP)
- Multiple Industrial Sectors

Architecture:

- Master-slave paradigm
- Master polls RTUs for data
- RTUs can send unsolicited messages
- DNP3 messages encapsulated in TCP/IP

Applications: Power grids, water treatment, oil & gas pipelines, transportation, building automation, healthcare systems

Threat Model: Attack Types

Eight Distinct DNP3 Attacks Detected

1. **Replay Attack:** Retransmitting valid messages for unauthorized execution
2. **Disable Unsolicited Message:** Preventing critical event notifications
3. **Enumerate Attack:** Unauthorized system reconnaissance
4. **Info Attack:** Gathering sensitive system information
5. **Initialize Data Attack:** Forcing RTU configuration reset
6. **Stop Application Attack:** Terminating critical applications
7. **Cold Restart Attack:** Complete system reboot with data loss
8. **Warm Restart Attack:** Restart while maintaining session state

Research Gaps Identified

Limitations in Existing Work

1. **Limited Multi-Class Classification:** Most focus on binary (attack vs. normal)
2. **Insufficient Features:** Many use only 6-20 features
3. **Unrealistic Testing:** Balanced datasets not representative of real-world
4. **Lack of Comparative Analysis:** Limited algorithm comparison
5. **Dataset Limitations:** Proprietary or limited datasets

Our Approach

99 features, multi-class (9 classes), variable intrusion rates, 4 algorithms compared

Dataset Description

DNP3 Intrusion Detection Dataset

- **Source:** Radoglou-Grammatikis et al. testbed
- **Total Records:** 365,099 network flow instances
- **Collection Period:** May 14-19, 2020
- **Attack Duration:** 4 hours per attack type
- **Network Topology:** 8 industrial entities, 1 HMI, 3 intruder machines
- **Initial Features:** 105 (reduced to 99 after selection)
- **Classes:** 8 attack types + 1 normal = **9 classes**

Tools Used: Nmap, Scapy (traffic capture), CICFlowMeter (TCP/IP features), custom DNP3 parser

Implementation Pipeline - Overview

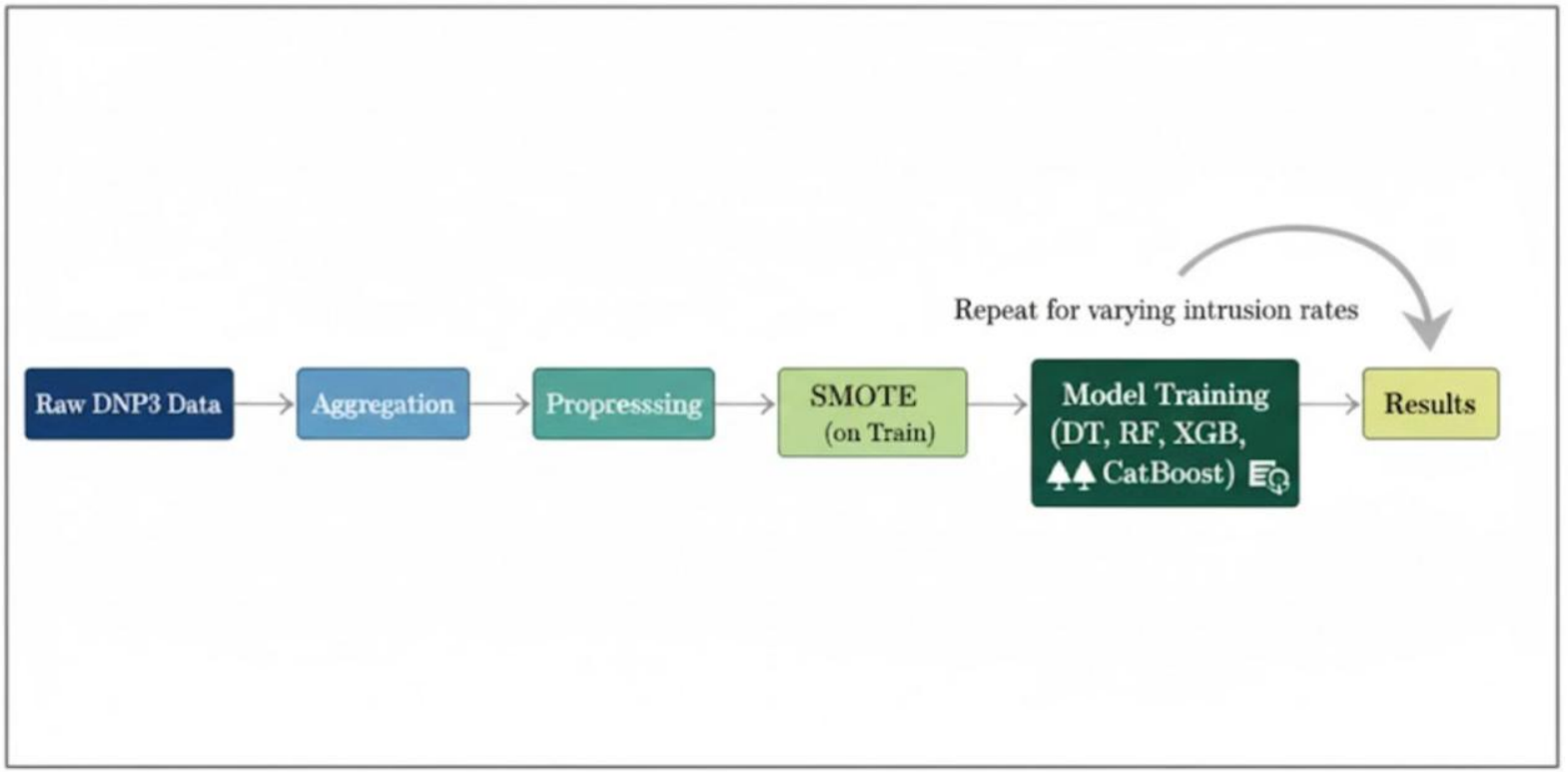

Figure: IDS Implementation Pipeline

Phase 1: Data Aggregation

Goal

Combine separate CSV files into one labeled master dataset

Process:

1. Process attack folders - label each file with attack type
2. Process training data containing BENIGN traffic
3. Concatenate all dataframes
4. Save unified master dataset

Output: Single CSV with all attack types and normal traffic properly labeled

Phase 2: Data Preprocessing

Transformation Summary

Metric	Before	After
Total Rows	378,419	378,419
Total Columns	188	183
Data Types	Mixed (12 object)	Fully Numeric
Missing Values	33M+	0
Infinite Values	10	0

Key Steps:

- Clean column names, encode labels
- Convert object columns (IPs, timestamps) to numeric
- Fill NaN and replace inf with 0
- Drop unnecessary identifier columns

Phase 3: Feature Selection

Goal

Reduce to 99 most informative features for model efficiency

Process:

1. **Scale Data:** StandardScaler (mean=0, std=1)
2. **Remove Low Variance:** VarianceThreshold filter
3. **Analyze Structure:** PCA (insight only - not used for transformation)
4. **Rank Features:** SelectKBest with ANOVA F-test
5. **Select Top 99:** Highest scoring features

Rationale: Maintain interpretability, align with baseline research

Training Strategy

Four ML Algorithms Tested

- **Decision Tree** - Single tree, Gini criterion
- **Random Forest** - 100 trees, bootstrap sampling
- **XGBoost** - Gradient boosting, mlogloss metric
- **CatBoost** - Gradient boosting with categorical handling

Data Split

70/30 stratified split - Training (70%), Testing (30%)

Intrusion Rate Scenarios:

- 5% attacks, 95% normal
- 10% attacks, 90% normal
- 15% attacks, 85% normal

Handling Class Imbalance

SMOTE Oversampling

Synthetic Minority Over-sampling Technique

- Generates synthetic samples by interpolation
- Balances all 9 classes to equal representation
- Prevents model bias toward majority (normal) class
- Avoids simple duplication that causes overfitting

Example (5% scenario):

- Initial: 10,000 normal + 500 attacks
- After SMOTE: 10,000 normal + 10,000 attacks (synthetic)
- Result: Balanced 20,000-sample training set

Implementation Environment

Software:

- Windows 11, 64-bit
- Python 3.8+
- Google Colab

Hardware:

- Intel Core i5-1135G7
- 8 GB RAM
- NVIDIA GeForce MX450

Libraries:

- pandas, numpy
- scikit-learn
- xgboost, catboost
- imbalanced-learn
- matplotlib, seaborn

Training Time: Decision Tree/Random Forest (seconds), XGBoost/CatBoost (minutes)

Performance Metrics - All Algorithms

Algorithm	Intrusion Rate	Accuracy	Precision	Recall	F1-Score
3*XGBoost	5%	**0.9403**	0.9409	0.9403	0.9382
	10%	**0.9450**	0.9451	0.9450	0.9436
	15%	**0.9485**	0.9485	0.9485	0.9474
3*CatBoost	5%	0.9245	0.9245	0.9245	0.9233
	10%	0.9353	0.9353	0.9353	0.9344
	15%	**0.9397**	0.9396	0.9397	0.9389
3*Random Forest	5%	0.9290	0.9299	0.9290	0.9250
	10%	0.9392	0.9395	0.9392	0.9369
	15%	**0.9427**	0.9428	0.9427	0.9411
3*Decision Tree	5%	0.9312	0.9313	0.9312	0.9271
	10%	0.9356	0.9359	0.9356	0.9332
	15%	**0.9398**	0.9397	0.9398	0.9377

Best Model: XGBoost at 15% intrusion rate - 94.85% accuracy

Key Observations

Effect of Intrusion Rate

- Accuracy improved slightly from 5% to 15% (0.5-1.5 percentage points)
- At 5%, some rare attack types completely absent from training
- Models handled imbalance well due to SMOTE

Algorithm Ranking

1. **XGBoost** - 94.85% (best)
2. **CatBoost** - 93.97%
3. **Random Forest** - 94.27%
4. **Decision Tree** - 93.98% (lowest)

Gradient boosting methods outperformed single tree and random forest

Comparison with Original Study

Study	XGBoost Accuracy	Gap
Original (Dangwal et al. 2024)	99.56%	–
Our Replication	94.85%	-4.71%

Potential Reasons for Gap

- **Overfitting:** Original may have overfit training data
- **Data Leakage:** Possible cross-validation inflation
- **Hyperparameter Tuning:** Original likely optimized parameters
- **Different Test Methodology:** We used strict 30% holdout

Our Interpretation

94.85% is more realistic, conservative, and generalizable for deployment

Positive Findings

Validation of Approach

- **Confirmed relative effectiveness:** XGBoost >CatBoost >Random Forest >Decision Tree
- **Excellent multi-class performance:** 94.85% for 9-class problem
- **Very low false positive rate:** <1% (critical for SCADA)
- **Robust under imbalance:** Performance degradation minimal at 5% intrusion rate

Practical Implications

- Suitable for real-world deployment
- Can identify specific attack types for targeted response
- Maintains accuracy even with rare attacks

Security Efficacy

Strengths

- **Multi-class detection:** Identifies which specific attack is occurring
- **Near real-time:** Fast inference (milliseconds per input)
- **Broad coverage:** Detects 8 distinct DNP3 attack types
- **Low false positives:** Minimal alert fatigue
- **Passive monitoring:** Does not interfere with operations

Limitations

- **Zero-day attacks:** Cannot detect novel, unseen attack patterns
- **Evasion potential:** Sophisticated adversaries may mimic normal patterns
- **Network-layer only:** Does not monitor host-level threats
- **False negatives:** Some attacks may be missed (safety concern)

Future Research Directions

1. **Unsupervised Anomaly Detection:** Integrate autoencoder for zero-day attacks
2. **Feature Optimization:** Investigate if $<$99 features achieve similar results (SHAP analysis)
3. **Deep Learning Models:** RNNs, TCNs for time-series pattern capture
4. **Cross-Domain Validation:** Test on different SCADA testbeds and protocols (Modbus)
5. **Real-time Prototype:** Live DNP3 network integration with automated response
6. **Advanced Imbalance Techniques:** ADASYN, weighted loss functions
7. **Explainability:** LIME/SHAP integration for operator trust
8. **Cyber-Physical Fusion:** Combine network IDS with physical sensor anomaly detection

Conclusion

Key Achievements

- Successfully replicated DNP3 IDS with 94.85% accuracy (9-class)
- Validated machine learning effectiveness for SCADA security
- Demonstrated robustness under realistic imbalanced scenarios
- Comprehensive comparison of 4 algorithms across 3 intrusion rates

Contributions

- More conservative, realistic performance assessment
- Detailed security and complexity analysis
- Practical deployment considerations for SCADA-IoT
- Open pathway for future enhancements

Conclusion: Machine learning-based IDS is a viable, practical solution for protecting DNP3 SCADA systems from cyber intrusions.

References I

[1] Keshk, M., et al. (2017). “Privacy-Preserving Intrusion Detection Technique for SCADA Systems.” *IEEE Conference Proceedings*.

[2] Al-Asiri, M., et al. (2020). “Physical Process-Based Intrusion Detection for SCADA Systems.” *Journal of Critical Infrastructure Protection*.

[3] Radoglou-Grammatikis, P., et al. (2020). “DIDEROT: An Intrusion Detection and Prevention System for DNP3-based SCADA Systems.” *ACM Conference Proceedings*.

[4] Altaha, M., & Hong, J. (2023). “FC-AE-IDS: Anomaly-Based IDS for DNP3 Function Codes.” *IEEE Transactions on Industrial Informatics*.

References II

[5] Diab, A., et al. (2023). “Genetically-Seeded Flora Feature Optimization with Transformer Neural Networks for SCADA IDS.” *WUSTL-IIoT-2018 Dataset Study*.

[6] Mesadieu, F., et al. (2024). “Deep Reinforcement Learning for SCADA Intrusion Detection.” *Industrial Control Systems Security Journal*.

[7] Kelli, V., et al. (2022). “DNP3-Specific Deep Neural Networks for Intrusion Detection.” *Protocol Security Symposium*.

[8] Dangwal, R., et al. (2024). “IDM-DNP3: Multi-Class Machine Learning-Based Intrusion Detection for DNP3 SCADA Systems.” *International Journal of Critical Infrastructure Protection*.

Thank you very much!

Questions?

V S M Bharat Kumar - 2024202030
Sai Charan Thammi - 2024202021
Aditya Sangana - 2024202027