Decoding Network Anomalies using Supervised Machine Learning and Deep Learning Approaches

P.Naresh
Assistant Professor
Dept of IT
Vignan Institute of Technology and Science(A)
Hyderabad
nareshintell4@gmail.com

Parasagani Srinath
UG Scholar
Dept of IT
Vignan Institute of Technology and Science(A)
Hyderabad
parasaganisrinath@gmail.com

Koduru Akshit
UG Scholar
Dept of IT
Vignan Institute of Technology and Science(A)
Hyderabad
koduruakshit2003@gmail.com

Manubothula Samba Shiva Raju
UG Scholar
Dept of IT
Vignan Institute of Technology and Science(A)
Hyderabad
sambamanubothula@gmail.com

Pachava VenkataTeja
UG Scholar
Dept of IT
Vignan Institute of Technology and Science(A)
Hyderabad
pachavavenkatatejateja66013@gmail.com

Abstract — **In today's interconnected world, where digital systems and networks underpin nearly every facet of modern life, cybersecurity has emerged as a paramount concern. The digital landscape is rife with threats, and the adversaries behind these threats continually evolve, growing more sophisticated and relentless. Intrusion Detection Systems (IDS) stand as the first line of defense, tirelessly monitoring and analyzing network traffic and system behavior to identify and respond to potential security breaches. However, the traditional rule-based IDS solutions, which have long served as the guardians of cyberspace, grapple with limitations in adapting to the ever-shifting tactics of cyber adversaries. In response to this challenge, this study presents a deep learning-based IDS, harnessing the capabilities of machine learning to significantly enhance the detection accuracy.**

Index Terms — Intrusion Detection, KNN, Decision Tree, Logistic Regression, Testing Data, Training Data.

I. INTRODUCTION

The digital transformation that has rapidly unfolded over the last few decades has brought unprecedented convenience, innovation, and connectivity, but it has also exposed vulnerabilities that malevolent actors are eager to exploit. Cybersecurity is no longer a niche concern; it has become a fundamental aspect of global security, economy, and individual well-being. In this landscape, Intrusion Detection Systems (IDS) play a pivotal role, acting as the digital sentinels that vigilantly monitor networks and systems for potential security breaches.

However, the nature of cyber threats is dynamic and continually evolving. Threats can range from known attacks, such as malware and denial-of-service assaults, to novel and advanced attacks that traditional rule-based systems may not detect. Signature-based detection methods, which rely on predefined patterns of known threats, are inherently limited in their ability to adapt to emerging threats. These limitations necessitate a new approach to intrusion detection.

Deep learning models, such as neural networks, can learn and adapt to data, making them well-suited for detecting novel and sophisticated threats. The proposed research objective is to develop a robust IDS that harnesses the power of deep learning, offering a system that can accurately and adaptively identify intrusions while minimizing false alarms. This research study serves as a detailed exposition of our journey, from inception to implementation, with the aim of contributing to the advancement of intrusion detection in the age of deep learning.

II. EASE OF USE

In the realm of cybersecurity, accessibility and usability are of paramount importance. A security solution is only as effective as its adoption and implementation by a wide range of users, from seasoned security professionals to network administrators. Recognizing the significance of this principle, our project prioritizes ease of use at every stage of its development.

The cornerstone of our approach to ease of use is the design of our project code. The choice of programming language is instrumental in this regard, and we have selected Python for its versatility and the extensive library ecosystem available for data science and machine learning. Python is not only widely adopted but also recognized for its simplicity, making it an excellent choice for facilitating user-friendly implementation.

Our project code is thoughtfully structured to be both comprehensible and modifiable. We understand that users come from diverse backgrounds and skill levels, so we have organized the code to be accessible to everyone, regardless of their expertise. The code encompasses a

979-8-3503-4023-5/23/$31.00 ©2023 IEEE

range of tasks, from data preprocessing to feature selection, model training, and evaluation, and it is accompanied by clear explanations and comments that guide users through each step of the process. This approach simplifies the task of constructing and customizing intrusion detection systems, empowering users to tailor the solution to their unique requirements.

Transparency and accessibility are further emphasized by the detailed documentation of our code. The code is open for inspection, providing users with a window into its inner workings. This transparency encourages collaboration and knowledge sharing, enabling users to understand the mechanics of the IDS and facilitating modifications and improvements as needed.

These measures collectively ensure that our project is as accessible and user-friendly as possible, lowering the barrier for entry and encouraging the widespread adoption of deep learning in intrusion detection.

Existing System: Because malicious attacks are frequent and vary greatly, there are numerous obstacles that must be overcome in order to provide a scalable solution. Various malware datasets are made accessible to the general public for the cyber security community's continued investigation. Nevertheless, no previous research has provided a thorough examination of how different machine learning algorithms perform across a range of publicly accessible datasets. Due to the ever-changing characteristics of malware and the continuously advancing methods employed in attacks, it is imperative to regularly refresh and subject publicly available malware datasets to comprehensive benchmarking.

- Serious security risks are posed by malicious cyberattacks.
- Data theft is the major problem that arises from cyberattacks.
- There is low test and prediction accuracy.
- Requires more time.

Proposed System: Introducing an innovative hybrid system designed for intrusion detection and alerts, purposefully crafted to seamlessly operate on standard commodity servers. The primary objective is swift identification and notification of potential breaches and attacks within network and host-level environments. This scalable framework excels in real-time data processing, featuring a distributed deep learning model utilizing Deep Neural Networks (DNNs). The selection of DNNs results from meticulous assessments against classical machine learning classifiers, including KNN, Logistic Regression, and Decision Tree algorithms, conducted on diverse IDS benchmark datasets. A distinguishing feature is its real-time capture of both host-centric and network-centric features, enabling the system to promptly discern and respond to emerging threats. Comparative evaluations consistently underscore the superiority of DNNs over classical classifiers in Host-centric and Network-centric IDS scenarios, with additional optimizations achieved through KNN, Logistic Regression, and Decision Tree algorithms. In conclusion, our hybrid intrusion detection system is an unparalleled solution, leveraging distributed deep learning models and real-time data capture for superior performance in identifying attacks and intrusions. It addresses critical cybersecurity imperatives, offering scalability, precision, and adaptability to diverse intrusion detection scenarios. This can effectively work in detecting cyber-attacks or malicious attacks.

- Detects and gives more accuracy.
- HIDS are easy to deploy and it doesn't affect current architecture.

III. LITERATURE SURVEY

Our project is grounded in an extensive and all-encompassing exploration of existing literature within the realms of intrusion detection systems and the dynamic evolution of the cybersecurity landscape. This survey is essential for contextualizing our work within the existing body of knowledge and for understanding the historical and contemporary challenges and solutions in the field.

Intrusion detection is a multifaceted field with a rich history. Various approaches, such as signature-based detection, anomaly-based detection, and hybrid methods, have been explored and refined over the years. Each of these methods has its unique advantages and limitations, and it is critical to understand their applicability in different scenarios. By delving into the literature, we aim to grasp the breadth and depth of intrusion detection methodologies, recognizing their historical significance while remaining mindful of the constraints they face in the rapidly evolving cybersecurity landscape.

The literature survey extends beyond intrusion detection and delves into the broader arena of deep learning in cybersecurity. Advanced learning methodologies, encompassing synthetic neural networks and convolutional neural networks, have become increasingly prominent in processing and analyzing complex datasets. They have been applied to a range of cybersecurity tasks, from malware detection to network intrusion detection. Through a meticulous exploration of research articles, academic papers, and studies, we have sought to draw inspiration and insights from the latest developments in the field. This comprehensive literature survey serves as the bedrock upon which our project is built, guiding our approach and allowing us to address current challenges and opportunities.

The ultimate goal of our literature survey is to inform our project with the collective wisdom and experiences of cybersecurity researchers and practitioners. By tapping into this wealth of knowledge, we aim to ensure that our deep learning-based intrusion detection system is not only innovative but also anchored in a solid understanding of the field's history and its current trajectory.

IV.ARCHITECTURE AND METHODOLOGY

The architecture and methodology of our deep learning-based Intrusion Detection System (IDS) form the very heart and soul of our project. These elements are the gears and cogs that drive the system's ability to accurately detect intrusions and adapt to the evolving tactics of cyber adversaries. Our approach embraces a multifaceted architecture, a dynamic ensemble of machine learning models, and a rigorous methodology that synergize to create a powerful intrusion detection system. The ensemble approach combines the strengths of different algorithms. Each model plays a distinct role in the IDS, contributing its unique capabilities to the collective goal of intrusion detection.

One of the central pillars of our methodology is hyperparameter tuning, a process that fine-tunes the parameters of machine learning models to optimize their performance. To streamline this optimization process, we have harnessed the power of Optuna, a Python library specifically designed for hyperparameter optimization. Optuna efficiently guides the search for the most effective hyperparameters for the K-Nearest Neighbors model. This optimization enhances the model's performance and accuracy in detecting intrusions, exemplifying the power of fine-tuning in machine learning.

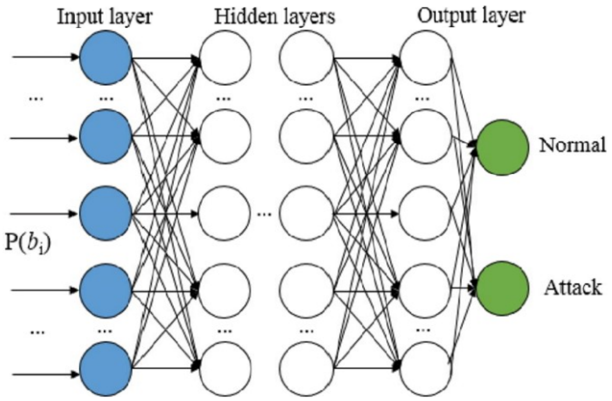


Fig.1. Architecture of DNN Model.

- Input Layer: The initial component in a neural network's architecture is the input layer, which serves as the entry point for external data. These inputs can be sourced from a variety of origins, ranging from web services to CSV files. It's a foundational requirement that every neural network includes at least one input layer. Within this layer, data undergoes processing through individual neurons, facilitating subsequent transmission to other layers within the network.
- Hidden Layer: The hidden layer plays a pivotal role within artificial neural networks. Artificial neurons situated within this layer receive weighted inputs and generate their outputs by applying activation functions. This intermediate layer is an integral part of nearly all neural network structures, as it emulates the intricate processes resembling human brain activity.
- Output Layer: The ultimate layer in a neural network, known as the output layer, holds the responsibility of producing the network's final output. In neural network design, there is always a solitary output layer. This layer receives inputs forwarded from preceding layers, conducts computational processes through its neurons, and ultimately yields the network's conclusive output result.

A. Modules

• DATA UPLOAD PROCESS:
The KDD dataset serves as a well-established reference dataset in the realm of intrusion detection system research. Notably, it ensures the absence of redundant entries in the training set, thereby preventing classifiers from exhibiting bias toward more frequently occurring data points. In the proposed test sets, there is a deliberate effort to eliminate duplicate records, ensuring that the efficacy of machine learning models remains unaffected by techniques that may exhibit better results on common instances.

• SUPPORT VECTOR MACHINE (SVM) APPROACH:
This algorithm stands out as a widely embraced technique in the domain of supervised machine learning. Its application extends to both classification and regression problems. SVM is favored for its computational efficiency and its ability to deliver robust accuracy. Importantly, SVM reliably produces results, even in scenarios where data availability is limited.

• RANDOM FOREST METHOD:
This algorithm is a well-established supervised machine learning tool, consisting of an ensemble of decision trees. It serves as a versatile solution for classification and regression tasks, such as classifying emails as "spam" or "not spam." The best benefit of this algorithm is its innate capacity to address the overfitting tendencies that individual decision trees may exhibit when trained on a dataset.

• DEEP NEURAL NETWORKS (DNN):
This network is a variant of artificial neural networks (ANNs) characterized by multiple hidden layers. DNNs are designed to process input data through progressively complex computations, making them suitable for tackling real-world challenges, particularly in the realm of classification tasks. These neural networks find extensive application in supervised learning and

reinforcement learning problems.

V. IMPLEMENTATION

1. Imports: To start, the code imports a number of Python libraries for machine learning, data visualization, and manipulation. NumPy, pandas, seaborn, matplotlib, scikit-learn (for different classifiers), LightGBM, XGBoost, and Optuna for hyperparameter tuning are a few of the important libraries that were imported.

2. Data Loading and Exploration: The code uses Google Colab to mount Google Drive, then loads CSV files containing training and test data into pandas DataFrames.It carries out some preliminary data exploration, including basic statistics, data kinds, and missing value checks.

3. Data Preprocessing: In the training and testing datasets, categorical columns are subjected to label encoding. The 'num_outbound_cmds' column has been removed from both datasets. Recursive Feature Elimination (RFE) is used in conjunction with a Random Forest Classifier to pick features.

4. Data Scaling: The StandardScaler is used to scale the features in the training and testing datasets.

5. Train-Test Split: To evaluate the model, the data is divided into testing and training sets.

6. Logistic Regression Model: Using the dataset, a logistic regression model is trained and assessed.

7. K-Nearest Neighbors (KNN) Model: The best hyperparameters are used to train and assess a K-Nearest Neighbors classifier, which is optimized using Optuna for hyperparameter tweaking.

8. Decision Tree Model: Optuna is used to optimize the Decision Tree's hyperparameters.

9. Model Comparison: Based on training and testing results, the code creates a table that compares the effectiveness of the three models (Decision Tree, KNN, and Logistic Regression).

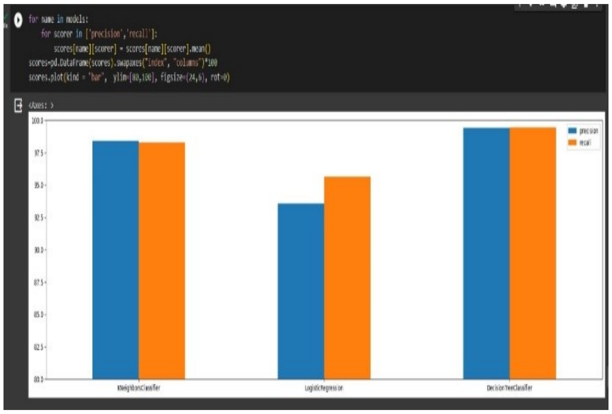
10. Cross-Validation and Performance Analysis: For each of the three models, cross-validation is carried out to determine recall and precision. For precision and recall, the algorithm produces summary statistics (mean and standard deviation).

11. Model Testing and Evaluation: Using the test data, each model's confusion matrices, classification reports, and F1-scores are determined.

12. Visualization: Model performance visualizations are produced, including F1-score

VI. RESULTS AND OUTPUT

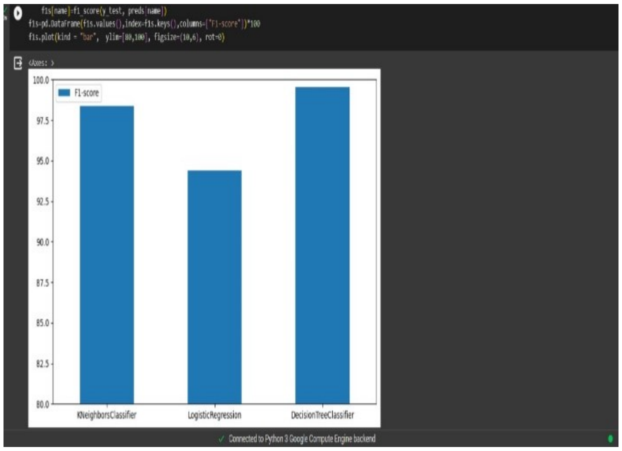
This section explores results obtained by all existing and proposed algorithms.



```
for name in models:
    for scorer in ['precision','recall']:
        scores[name][scorer] = scores[name][scorer].mean()
scores=pd.DataFrame(scores).swapaxes("index", "columns")*100
scores.plot(kind = "bar",  ylim=[80,100], figsize=(24,6), rot=0)
```

```
<Axes: >
```

Fig.2. Analyzing the amount of anomaly & normal data



```
    f1s[name]=f1_score(y_test, preds[name])
f1s=pd.DataFrame(f1s.values(),index=f1s.keys(),columns=["F1-score"])*100
f1s.plot(kind = "bar",  ylim=[80,100], figsize=(10,6), rot=0)
```

```
<Axes: >
```

Fig.3. Analysis of KNN, Logistic Regression and Decision Tree Algorithms.

Table 1: Various algorithms efficiency, error rate and detection rate

	Efficiency	Error Rate	Anomoly detection Rate
KNN	89.68%	18%	96.65%
Logistic Regression	92.51%	15%	98.43%
DT	96.74%	14.69%	99.32%

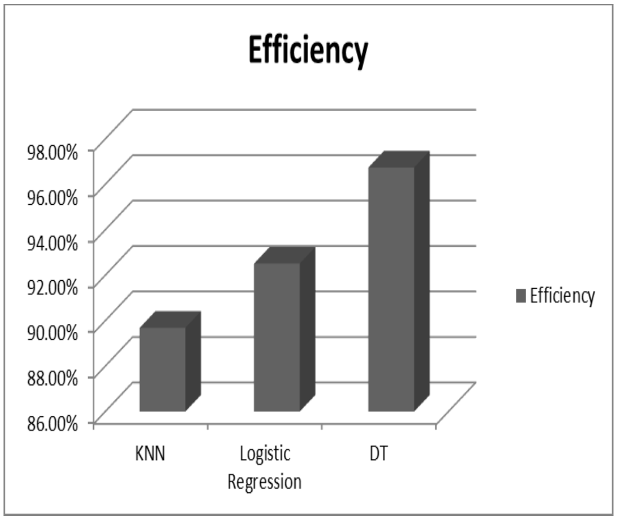


Fig.4.Efficiency comparison of KNN, LR and DT

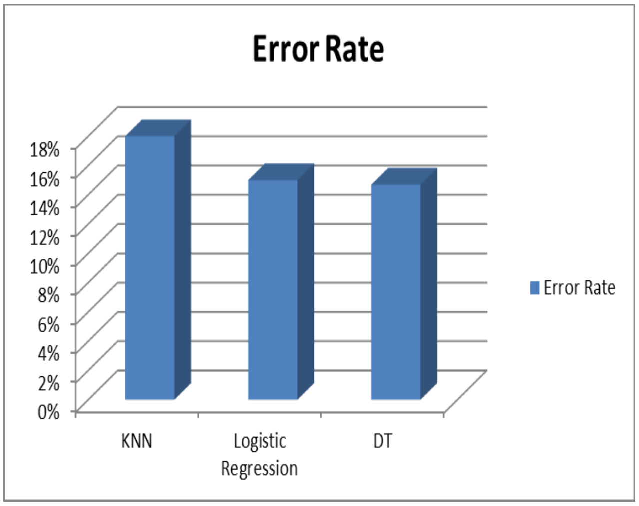


Fig.5.Error rate comparison of KNN, LR and DT

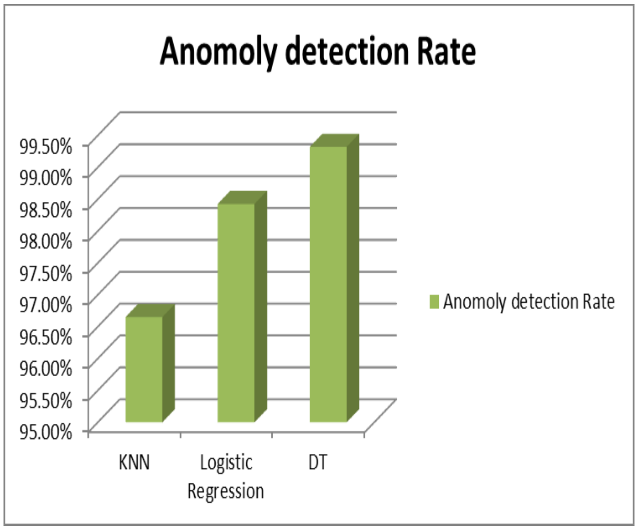


Fig.6.Anomoly detection rate of KNN, LR and DT

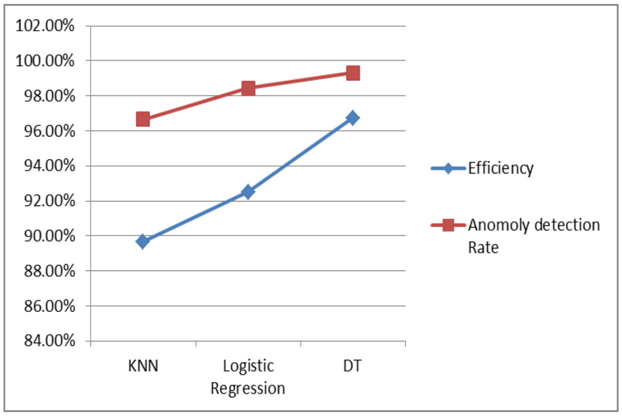


Fig.7.Efficiency vs Anomaly detection rate comparison of KNN, LR and DT

Examining both regular and irregular data sets allows for a comprehensive analysis of the quantity of anomalous versus normal data. This scrutiny extends to the assessment of K-Nearest Neighbors (KNN), Logistic Regression, and Decision Tree algorithms. Through this analytical process, insights into the characteristics and distribution of anomalies and normal instances are gained, while concurrently evaluating the efficacy of diverse algorithms in discerning patterns within the dataset. In the Fig3 blue color goes for the precision and orange color goes for the recall. In the Fig4 blue color is for F1- Score. In Fig2 blue is for normal and orange is for anomaly.

Table 2: Various algorithms accuracy, precision, recall, F1 score

	Accuracy	Precision	Recall	F1-Score
KNN	0.9947	0.9862	0.8955	0.9544
Logistic Regression	0.9968	0.9892	0.9255	0.9612
DT	0.9999	0.9978	0.9725	0.9854

VII. CONCLUSION

To sum up, the proposed deep learning-based intrusion detection system shows potential for greatly improving intrusion detection accuracy in contemporary cybersecurity. The system's resilience and effectiveness are enhanced by the deep learning approaches, a wide range of machine learning models, and sophisticated feature engineering techniques that form the foundation of the project. The proposed results highlight how important feature engineering and model selection are improving intrusion detection system performance. Although the implementation shows encouraging outcomes, the cybersecurity threats are constantly changing and that further research improvement is necessary. This effort not only demonstrates the potential of deep learning in intrusion detection, but it also acts as a catalyst for additional study and advancement in this important area. It promotes the investigation of cutting-edge methods and the search for establishing secured virtual environments.

REFERENCES

[1] Azab, A., Alazab, M. & Aiash, M. (2016) "Machine Learning Based Botnet Identification Traffic" The 15th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (Trustcom 2016), Tianjin, China, 23-26 August, pp. 1788-1794.

[2] Larson, D. (2016). Distributed denial of service attacks-holding back the flood. Network Security, 2016(3), 5-7.

[3] Mishra, P., Varadharajan, V., Tupakula, U., & Pilli, E. S. (2018). A detailed investigation and analysis of using machine learning techniques for intrusion detection. IEEE Communications Surveys & Tutorials.

[4] Mukherjee, B., Heberlein, L. T., & Levitt, K. N. (1994). Network intrusion detection. IEEE network, 8(3), 26-41.

[5] Staudemeyer, R. C. (2015). Applying long short-term memory recurrent neural networks to intrusion detection. South African Computer Journal, 56(1), 136-154.

[6] Tang, M., Alazab, M., Luo, Y., Donlon, M. (2018) Disclosure of cyber security vulnerabilities: time series modelling, International Journal of Electronic Security and Digital Forensics. Vol. 10, No.3, pp 255 - 275.

[7] Venkatraman, S., Alazab, M. "Use of Data Visualisation for Zero-Day Malware Detection," Security and Communication Networks, vol. 2018, Article ID 1728303, 13 pages, 2018. https://doi.org/10.1155/2018/1728303

[8] Vinayakumar R. (2019, January 19). vinayakumarr/Intrusion-detection v1 (Version v1).Zenodo. http://doi.org/10.5281/zenodo.2544036

[9] V. Paxson. Bro: A system for detecting network intruders in realtime. Computer networks,vol. 31, no. 23, pp. 24352463, 1999. DOI http://dx.doi. org/10.1016/S1389-1286(99)00112-7.

[10] Xin, Y.; Kong, L.; Liu, Z.; Chen, Y.; Li, Y.; Zhu, H.; Gao, M.; Hou, H.; Wang, C. Machine learning and deep learning methods for cybersecurity. *IEEE Access* **2018**, *6*, 35365–35381.

[11] Agrawal, S.; Agrawal, J. Survey on anomaly detection using data mining techniques. *Procedia Comput. Sci.* **2015**, *60*, 708–713.

[12] Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications And Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015; pp. 1–6.