#### **IFT 525: AI in Cybersecurity**

#### **AI-Powered Traffic Monitoring for Supply Chain and MITM Attack Detection**

#### **Information Technology, Arizona State University**

#### **Dr. Sami Mian**

#### **04/30/2025**

#### **Hemanth Kumar Guntupalli, Pooja Thalur**

#### **Abstract**

Supply chain cybersecurity has become a major concern, especially with how much modern software depends on third-party tools and services. Traditional methods like static scans and vendor checks often miss sophisticated threats — especially those hiding behind trusted vendors or using encrypted traffic to stay undetected. These approaches don’t offer the real-time awareness needed to catch today’s fast-changing risks.

To tackle this, we designed an AI-powered web traffic monitoring system that uses unsupervised anomaly detection, explainable AI, and federated learning. It continuously monitors third-party activity without needing labeled data and keeps user data private through decentralized learning. With explainable AI built in, the system doesn’t just throw alerts — it actually explains why something seems suspicious, which helps analysts make better decisions.

The result is a more responsive and trustworthy way to catch threats early and strengthen supply chain security using a privacy-friendly, adaptive AI approach.

#### **Introduction**

In recent years, enterprises have faced a sharp rise in supply chain attacks, where hackers take advantage of weak points in third-party software or vendors to breach larger organizations (Musalmari, 2024). High-profile incidents like the SolarWinds and Kaseya breaches showed how a single compromised supplier can impact thousands of downstream clients (OPSdesign, 2024). These cases highlight how difficult it is to maintain full visibility across complex digital supply chains. Attackers are also getting more sophisticated, using methods like dependency hijacking and even hardware implants that easily slip past traditional security tools.

Man-in-the-middle (MITM) attacks are another concern, where attackers intercept communication between trusted parties and manipulate data, often without detection. These types of attacks threaten both confidentiality and the integrity of enterprise systems (Kandasamy & Roseline, 2025). Unfortunately, many organizations still rely on outdated security tools like signature-based intrusion detection systems (IDS), which struggle to catch novel threats and tend to flood teams with false alerts (Kandasamy & Roseline, 2025). Smaller businesses, in particular, may not have the staff or tools needed to monitor their full supply chain.

AI has the potential to close this gap. It can monitor live traffic, learn normal patterns, and flag unusual behavior in real time. Machine learning (ML) and deep learning make it possible to automate this process at scale, detecting subtle red flags that humans might miss. For example, the SolarWinds attack involved SUNBURST malware being inserted into Orion software updates which is trusted by about 18,000 customers. Once deployed, the malware allowed attackers to move laterally, observe internal processes, and spread quietly. What's more troubling is that the actual targets were often multiple steps removed from SolarWinds, showing how deep these breaches can go.

A similar situation unfolded during the Log4j vulnerability crisis in 2021. According to Google’s Open-Source Insights team, over 17,000 packages in Maven Central were indirectly affected — even if they didn’t directly depend on the flawed log4j-core package. This incident made it clear that software dependencies can create hidden risks across entire ecosystems.

AI offers more than just detection, it can help build transparency and trust. Tools like explainable AI (XAI) make security alerts easier for analysts to understand and act on, rather than relying on black-box models. In the future, concepts like AI Bills of Materials (AIBOMs) may become standard, helping organizations track how AI models are built and used. As threats become more advanced, we’ll need smarter, real-time defenses that not only detect attacks quickly but also protect user privacy and support accountability (Kandasamy & Roseline, 2025).

#### **Literature Review**

Sammak, R., Dunlap, T., and Williams, L. tackle the complex problem of software supply chain security by focusing on three major attack surfaces: vulnerabilities in code dependencies, weaknesses in build infrastructure, and human factors. To mitigate these risks, both the industry and academic communities have introduced various tools and frameworks such as Software Composition Analysis (SCA) tools, Software Bills of Materials (SBOMs), OpenSSF Scorecard, SLSA, in-toto, and The Update Framework (TUF). While these have helped improve visibility, the authors point out that managing and updating dependencies is still challenging due to inconsistent vulnerability data. Selecting safe and well-maintained libraries remains difficult, and legacy build systems are often opaque and hard to secure. Human error continues to be one of the most unpredictable and dangerous threats, as seen in the xz-utils attack—an event that would have bypassed most existing protections. The paper also highlights how the rise of Large Language Models (LLMs) introduces a new layer of risk, necessitating fresh threat models and tailored security strategies.

Another paper looks into how software professionals are using AI assistants like ChatGPT and GitHub Copilot for security-related tasks, and what risks this might pose. Based on 27 interviews with developers and analysis of 190 Reddit posts, the study shows that these tools are already being used for secure code generation, threat modeling, code review, and vulnerability detection. Although widely adopted, most professionals are skeptical and still verify AI-generated output like they would a human peer. However, many expect reliance on AI to grow as the tools mature. The authors identify a key gap: developers often think they can accurately assess AI-generated code, but they may overestimate their ability to catch subtle security flaws. This mismatch could lead to vulnerabilities being missed. The study calls for stronger safeguards in how AI outputs are handled and stresses the need for more research on how these tools are reshaping secure software development (Klemmer, J. H., Horstmann, S. A., Patnaik, N., Ludden, C., Burton Jr., C., Powers, C).

A separate study evaluates whether open-source LLMs can effectively identify programming errors and deprecated code, potentially replacing static and dynamic security scanners. The results are encouraging but mixed. LLMs show promise in catching security-relevant issues, but they struggle with high memory loads and unpredictable inputs. The paper also points out a new concern: unconstrained prompts could be used to bypass security by sneaking in malicious code. While the authors believe LLMs have value as a complementary tool, they argue that serious limitations remain. These include difficulties in securely integrating new data, vulnerability to prompt manipulation, and the need for more critical oversight when deploying AI in sensitive environments (Alevizos, V., Papakostas, G. A., Simasiku, A., Malliarou, D., Messinis, A., Edralin, S., Xu, C., & Yue, Z.).

Ali and Abbas address how AI is being adopted in finance and software supply chains while maintaining data privacy and security. The paper discusses how tools like differential privacy, secure multi-party computation, anomaly detection, and secure code analysis can improve security without compromising operational performance. In financial markets, for example, AI can enhance decision-making but also risks leaking sensitive data. In the software supply chain, AI tools can be used to detect vulnerabilities and protect intellectual property throughout the development process. The authors warn that organizations must strike a balance between efficiency and data protection. Without that, the benefits of AI could be overshadowed by security breaches or loss of user trust (Ali, K., Abbas, U., & Bad, S.).

Finally, a study by Jiang and colleagues dives into the overlooked risks in pre-trained model (PTM) supply chains, particularly those distributed through popular model hubs. As developers increasingly reuse these models to save on training costs, the hubs themselves have become high-value targets. Through analysis of eight model hubs, the authors found that current defenses were often weak, leaving room for sophisticated attacks to go undetected. The study draws parallels between PTM security and traditional software supply chains, stressing the need for better threat modeling, auditability, and security tooling to protect these increasingly critical AI assets (Jiang, W., Synovic, N., Sethi, R., Indarapu, A., Hyatt, M., Schorlemmer, T. R., Thiruvathukal, G. K., & Davis, J. C.).

#### 

#### 

#### 

#### 

#### 

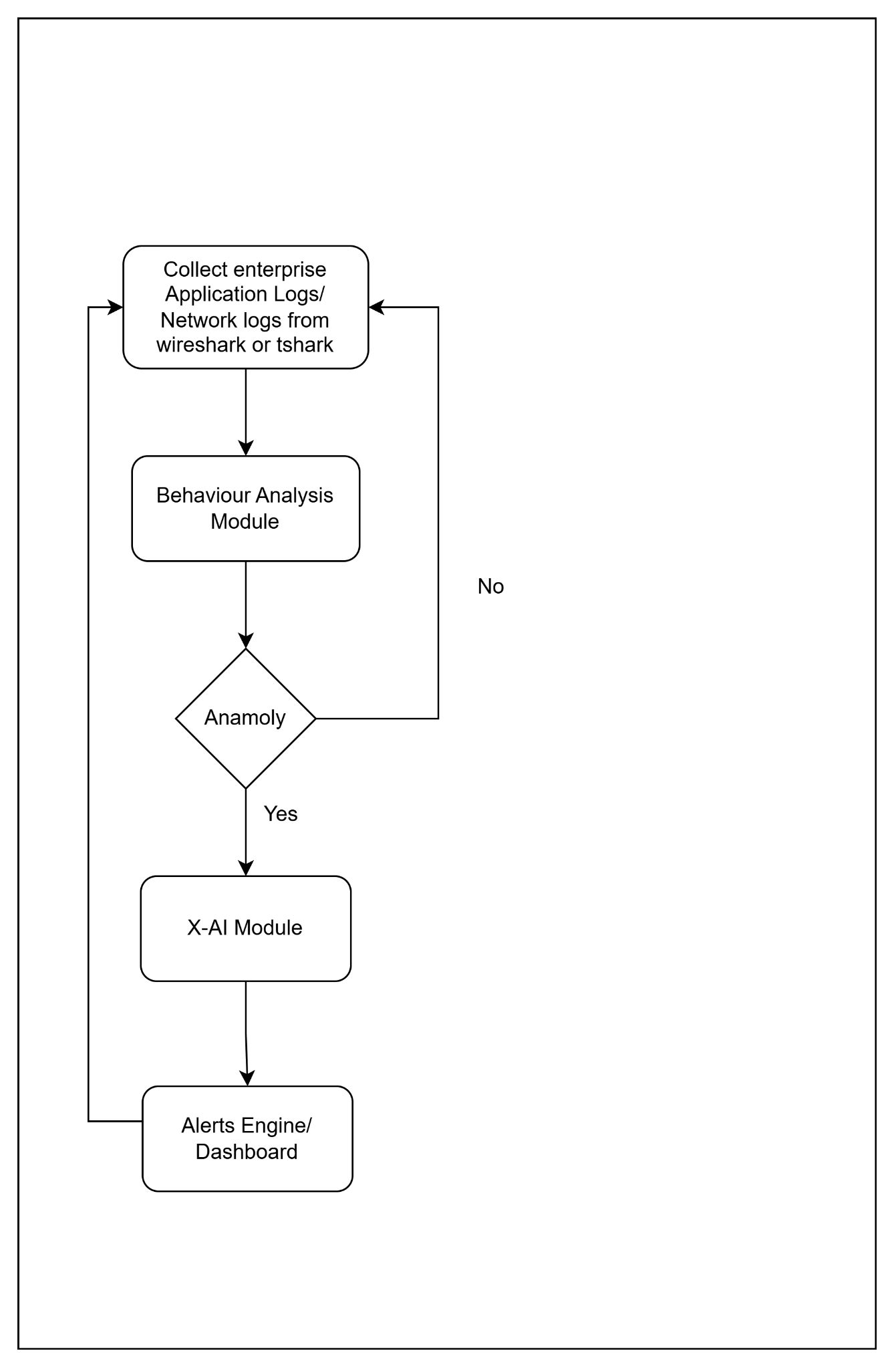
#### **Proposed Solution**

Our solution is a modular, AI-driven real-time web traffic monitoring system to detect supply chain exfiltration threats. The proposed end-to-end system operates through the following main components: Input Data sources, Data Preprocessing, Anomaly Detection Module, Explainable AI Module, Alerting and Response System.

1. Input Data Sources – The system starts by collecting data from a variety of sources. This includes web application that track how users and services interact with the system, network traffic logs such as packet captures and NetFlow records that show details about data moving through the network, internal databases that hold configuration files, access logs, and software inventories, and external intelligence feeds that provide up-to-date information about known threats and vulnerabilities.
2. Data Ingestion & Preprocessing - Once the data is collected, it's brought into the system using tools like Apache Kafka or AWS Kinesis, which can handle fast-moving, real-time information. The raw data is then cleaned and organized through feature extraction pipelines that give out useful details like IP addresses, the amount of traffic, or how APIs are being used. An anonymization layer is also applied to hide any sensitive or personal data, ensuring privacy is maintained**.**
3. Behaviour Analysis Module - With the data ready, the system analyzes it to look for unusual behavior. It uses unsupervised learning methods such as OC SVM or Isolation Forest, which don’t need pre-labeled examples, to find patterns that seem out of the ordinary. An explainable AI layer is added on top, using tools like LIME, ELI5 or SHAP to help security analysts understand why a certain behavior was flagged. There is also a federated learning component, which monitors third-party vendors while keeping their sensitive data decentralized and private.
4. Alerting and Response System **-** When the system detects something suspicious, it alerts the security team through a web dashboard that displays the findings in a clear and visual way. It also includes APIs that let it connect easily to existing security tools like Splunk or the Elastic Stack. Finally, there's a feedback loop where human analysts review the alerts, confirm whether they are valid, and their feedback helps the system improve its accuracy over time.

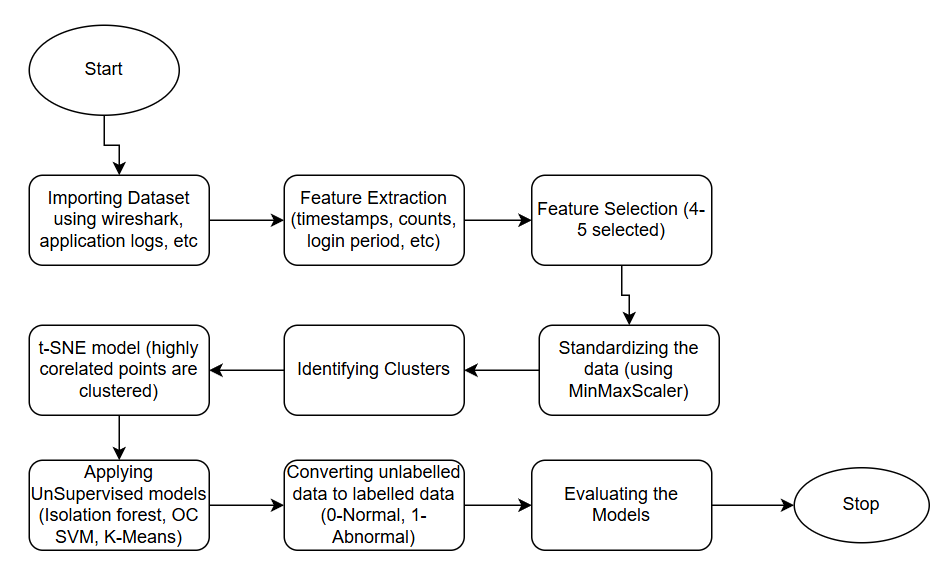
The below diagram (Figure 1) shows the overview of the described components.

**Solutions Architect-Level Diagram**



*Figure 1: The figure gives the high-level data flow of the system*

This diagram Figure 2, represents a complete pipeline for building an unsupervised anomaly detection system using network or log data. The process begins with importing the dataset from sources like Wireshark captures or application logs. Once the data is collected, relevant features such as timestamps, event counts, and login periods are extracted. From these, a smaller subset of the most useful features is selected ideally four to five. The selected data is then standardized using a MinMaxScaler to ensure all features are on the same scale, which is important for clustering and model performance. Clustering techniques are applied to identify patterns or groupings in the data, and a t-SNE model is used to visually cluster similar points together in a lower-dimensional space. After clustering, unsupervised learning models such as Isolation Forest, One-Class SVM, and K-Means are applied to detect abnormal behavior. The model outputs are then converted into labeled data where ‘0’ represents normal activity and ‘1’ represents anomalies. Finally, the labeled data is used to evaluate the performance of the models, completing the anomaly detection pipeline.

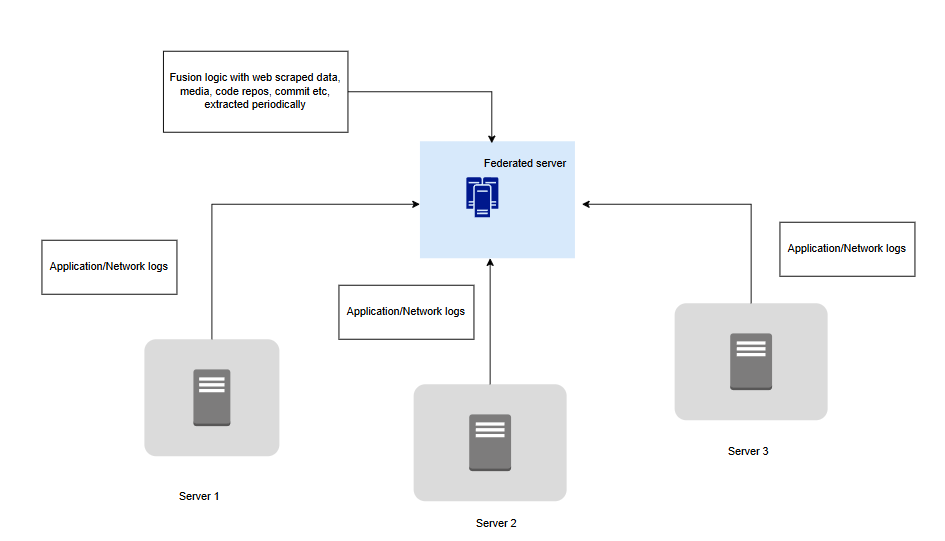
****

*Figure 2: The figure shows the workflow for Unsupervised Anomaly Detection Using Network and Log Data*

**Justification & Innovation:**

All data, including logs and model checkpoints, is securely stored in encrypted cloud storage solutions such as AWS S3 or Azure Blob. Real-time behavioral monitoring enables the detection of supply chain attacks that traditional Software Bill of Materials (SBOMs) and static vulnerability scanners often overlook. The use of explainable AI ensures that the decisions made by the system are transparent and understandable, fostering greater trust among analysts. Through federated learning, the system can continuously learn about third-party behavior across different organizations without centralizing sensitive data, thereby minimizing privacy risks. Additionally, the solution maintains low false positive rates by using intelligent clustering techniques, which significantly reduce alert noise compared to rigid, rule-based systems.

Compared to traditional static analysis or periodic audits, this solution proactively detects live attack attempts even in encrypted traffic and hence closing a major industry gap.



*Figure 3: Federated Learning Architecture for Privacy-Preserving Threat Detection with Integrated External Intelligence Fusion*

**Technical Details**

The Figure 3, represents a federated learning architecture used for secure and privacy-preserving threat detection or anomaly analysis across distributed environments. At its core, it includes multiple edge nodes—labelled as Server 1, Server 2, and Server 3—that independently collect and process application and network logs. These logs may include data such as user interactions, access events, network traffic, and security incidents. Instead of transmitting raw log data to a centralized location, each server locally trains its own model using the captured data, which preserves privacy and reduces the risk of data leakage.

The center of the diagram shows a federated server, which acts as a central coordinator in the learning process. Rather than collecting data, it receives model updates (such as weights or gradients) from each server. These updates are aggregated to build a more robust and generalized global model. The federated server then shares the improved model back with the individual servers, ensuring that each node benefits from collective insights without ever sharing its raw data. This cycle repeats periodically, improving detection accuracy while respecting data boundaries.

Above the federated server is a fusion logic component, which integrates external, periodically gathered intelligence. This includes web-scraped data, media articles, open-source code repositories, and commit histories. By combining this external data with insights from internal logs, the federated server gains a broader contextual view of potential threats and emerging vulnerabilities. This fusion layer enhances the overall learning process, allowing the system to adapt dynamically to new and evolving attack vectors.

Overall, this architecture is well-suited for organizations that require scalable, collaborative security monitoring while maintaining strict control over sensitive data. It provides a way to harness collective intelligence across distributed systems without sacrificing privacy, making it ideal for modern enterprise or cross-organization threat detection use cases.

**Ethical Design Considerations**

The system is designed to ensure fairness, privacy, and reliability in anomaly detection. To mitigate bias, models are trained on diversified and regularly rotated datasets, preventing the overflagging of specific vendors, locations, or patterns. Privacy is protected through the use of federated learning, which keeps data decentralized, and by applying strong anonymization techniques to prevent any exposure of sensitive client or vendor information. The system also emphasizes explainability, requiring that only transparent and interpretable models are used for generating alerts, allowing AI decisions to be reviewed and audited. To defend against manipulation, the models undergo regular adversarial testing to ensure they remain robust against sophisticated evasion techniques. Additionally, human oversight is built into the process, with a human-in-the-loop mechanism ensuring that high-severity alerts are always verified by analysts before any automated response is triggered.

**Validation Plan**

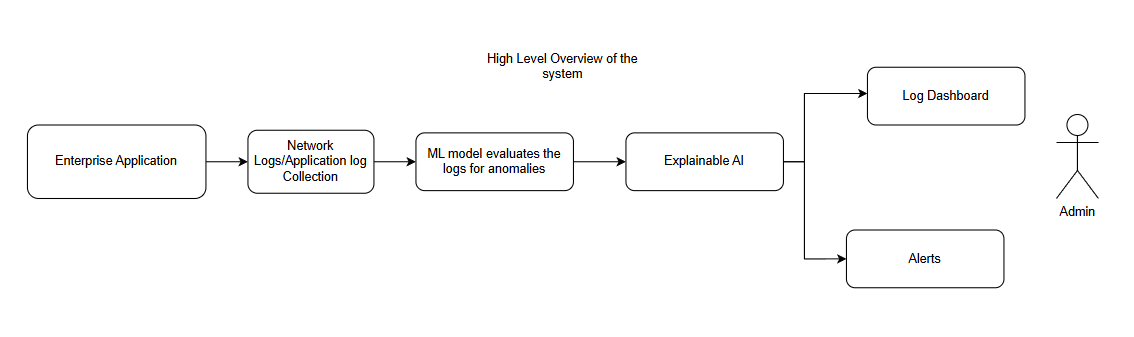
Our plan to test how well our AI system works and how reliable it is in real-world conditions includes using realistic data, checking performance with clear measures, using the right tools, and testing in environments that are similar to real companies.

* Dataset: We'll use fake but realistic network traffic that includes both normal behavior and known types of cyberattacks. We'll also use real logs from company networks (safely anonymized) to check how the system performs with actual traffic.
* Metrics: Accuracy, precision, recall, F1 score, latency (event-to-alert time), explainability ratings via human feedback. We’ll ask human analysts to rate how easy it is to understand the AI’s reasoning behind alerts. We will also measure the speed by monitoring how fast it detects and alerts on suspicious activity.
* Tools: We’ll use popular tools like Scikit-learn and TensorFlow to build the models. Tools like SHAP, LIME, and ELI5 will help explain what the AI is doing and why it made certain decisions. These tools help us build smarter and more understandable AI.
* Simulation Environment: We'll simulate a real business setup in the cloud to see how the system behaves in a live-like setting. We’ll also use local environments with virtual machines and containers to run more controlled tests. We’ll simulate real cyberattacks to test how the system reacts and adapts.

This approach ensures the system is accurate, fast, easy to understand, and useful in the real world. We’ll keep improving it based on testing and expert feedback.

#### **Discussion & Future Work**

Overall, the high level overview of the system can be seen from the figure below. Logs from an enterprise application are collected and analyzed by a machine learning model to detect anomalies. An explainable AI layer ensures the results are interpretable, which are then sent to a log dashboard and alert system which are later reviewed by an admin to take required actions.

****

*Figure 4: High-Level Architecture of an Explainable AI-Based Anomaly Detection System for Enterprise Logs*

Trade-offs exist between detection sensitivity and false positives. High model complexity might impact latency. Ethical concerns include ensuring continuous transparency and managing privacy while maintaining detection efficacy. Adversarial threats, where attackers try to poison training data or evade detection, must be continuously monitored.

Future improvements aim to strengthen the system by adding dark web scanning, CI/CD pipeline monitoring, continuous model updates based on analyst feedback, and enhanced defenses against evasion attacks. Overall, this work advances supply chain cybersecurity with a dynamic, explainable, and privacy-focused AI approach.

#### **References**

[Kandasamy, V., & Roseline, A. A. (2025). *Harnessing advanced hybrid deep learning model for real-time detection and prevention of man-in-the-middle cyber attacks. Scientific Reports, 15, Article 1697.*](https://doi.org/10.1038/s41598-025-85547-5)

[Musalmari, A. (2024, March 23). *Leveraging AI to defend against supply chain attacks. LinkedIn.*](https://www.linkedin.com/pulse/leveraging-ai-defend-against-supply-chain-attacks-musalmari-bzgcc)

Sammak, R., Dunlap, T., & Williams, L. (2025). Research directions in software supply chain security. ACM Transactions on Software Engineering and Methodology, 34(2), Article 12. <https://doi.org/10.1145/3714464>

Alevizos, V., Papakostas, G. A., Simasiku, A., Malliarou, D., Messinis, A., Edralin, S., Xu, C., & Yue, Z. (2024, December 26). *Integrating Artificial Open Generative Artificial Intelligence into Software Supply Chain Security*. arXiv. <https://arxiv.org/abs/2412.19088>​

Klemmer, J. H., Horstmann, S. A., Patnaik, N., Ludden, C., Burton Jr., C., Powers, C., Massacci, F., Rahman, A., Votipka, D., Lipford, H. R., Rashid, A., Naiakshina, A., & Fahl, S. (2024). *Using AI assistants in software development: A qualitative study on security practices and concerns*. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24)* (pp. 2726–2740). Association for Computing Machinery. <https://doi.org/10.1145/3658644.3690283>​

Ali, K., Abbas, U., & Bad, S. (2024, August). *AI solutions for enhancing data privacy in securities trading and software supply chains*. ResearchGate. <https://doi.org/10.13140/RG.2.2.12367.70569>

Jiang, W., Synovic, N., Sethi, R., Indarapu, A., Hyatt, M., Schorlemmer, T. R., Thiruvathukal, G. K., & Davis, J. C. (2022, November 11). *An empirical study of artifacts and security risks in the pre-trained model supply chain*. In *Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses (SCORED '22)* (pp. 105–114). Association for Computing Machinery. <https://doi.org/10.1145/3560835.3564547>​