DFA: Dynamic Frame Alteration For Video Manipulation Attack in IoT Environments

Buduka Cherish Nchelem
School of Computer science and Eletronic Engineering
University of Essex
Colchester, UK
bn23347@essex.ac.uk

Amit Kumar Singh
School of Computer science and Eletronic Engineering
University of Essex
Colchester, UK
a.k.singh@essex.ac.uk

Haralambos Mouratidis
School of Computer science and Eletronic Engineering
University of Essex
Colchester, UK
h.mouratidis@essex.ac.uk

Abstract—***The advent of Internet of Things (IoT) technology has revolutionized disaster management, providing real-time monitoring and enhanced response capabilities for critical situations like floods. Despite these advances, IoT systems are increasingly vulnerable to cyber-attacks, particularly data manipulation attacks that target video feeds. This paper presents a novel attack technique named Dynamic Frame Alternation (DFA) aimed at evading standard detection algorithms. Unlike traditional attacks like replay, frame injection, and video stream hijacking that modify frames in a linear or bulk approach, DFA strategically alters frames based on real-time changes in video attributes, such as color consistency and object presence. Leveraging metrics like the Structural Similarity Index Measure (SSIM) to detect optimal moments for frame manipulation, DFA enhances attack stealth by maintaining low detectability and resource usage. Experimental results, obtained from implementations on an embedded board platform, demonstrate that DFA consistently achieves lower detection rates when compared against traditional attacks while applying popular detection algorithms.***

Keywords— IoT Security, Real-Time Monitoring, Video Manipulation, DFA, Detection Algorithm

I. INTRODUCTION

Internet of Things (IoT) technologies have fundamentally transformed disaster management, particularly in areas vulnerable to natural calamities like floods [16]. IoT-based monitoring systems allow for real-time data acquisition from video feeds, sensor arrays, and other data points to support timely response and informed decision-making [14]. This capability is especially critical for flood-prone regions, where immediate access to environmental conditions can mitigate the impact of such events on affected communities. However, with the increased reliance on IoT infrastructure, new vulnerabilities have emerged. The complexity of IoT systems and their exposure to networks makes them prime targets for cyber-attacks, especially data manipulation attacks that compromise the integrity and reliability of video feeds and sensor data [21]. Consequently, the security of IoT systems in disaster management is a paramount concern, with a growing need for defenses against sophisticated attack methodologies.

Among various cyber-attacks, data manipulation attacks stand out for their potential to cause harm by tampering with the accuracy of information conveyed through video feeds. In flood monitoring systems, a compromised video feed can lead to inaccurate assessments of flood levels, hampering response strategies and endangering lives [10]. Traditional attacks like replay, frame injection, and video stream hijacking manipulate video content by injecting predefined or previously captured frames, disrupting the continuity of real-time footage. While effective, these attacks lack adaptability and are often detectable through common anomaly detection algorithms, which monitor abnormalities in frame sequences and computational patterns [22].

To advance attack while considering adaptability, attackers have developed more dynamic manipulation strategies. In this paper, we introduce a novel approach called dynamic frame alteration (DFA), designed to bypass traditional detection methods by adjusting the manipulation strategy based on real-time characteristics in the video feed. DFA exploits changes in video attributes, such as lighting, object presence, or scene stability, to trigger frame alterations precisely when it is least likely to be noticed. By doing so, DFA reduces detectability, creating a more stealthy and efficient form of attack that can evade standard anomaly detection tools. One of the primary innovations of DFA lies in its ability to determine the optimal moment for frame manipulation. Instead of relying on a static attack pattern, DFA employs real-time monitoring to identify contextual or visual cues in the video feed that signal an ideal manipulation window. Metrics like the Structural Similarity Index (SSIM) are employed to monitor for specific conditions, such as sudden changes in lighting or motion, which serve as triggers for the attack. This approach not only reduces the number of manipulated frames but also enhances the stealth of the attack, as alterations are applied only when the video feed naturally shifts, making them less noticeable to detection algorithms.

II. RELATED WORK

The study in [8] introduces mimic cloud ruling method aimed at defending against time-delayed covert channel attacks. The study first defines the attack model from the perspective of information theory, where attackers utilize response delay sample mean and variance as feature statistics to deduce the detection rate formula. To counter these attacks, the authors propose basic countermeasures and first-come-first-serve strategies designed to minimize the attackers' ability to exploit response time-delay differences for transmitting information. The authors in [9] designed an approach to detect and mitigate Cross-Site Scripting (XSS) attacks in IoT networks utilizing the Online Sequential Extreme Learning Machine (OS-ELM) algorithm, which suits the highly dynamic nature of IoT environments. This approach effectively handles real-time data and device heterogeneity by transforming attack vectors into a non-executable form, thus neutralizing the harmful effects of XSS attacks. This work provides a comprehensive classification of existing hardware attacks, with a focus on covert attacks. The authors propose a schema for quantifying these attacks and introduce various countermeasures to prevent them, highlighting the need for enhanced hardware security measures.

In [11] they investigate the impact of false data injection on IoT automotive engine sensors using several deep learning algorithms. The study finds that while SPNN excels in detecting continuous attacks, GAN is more effective for temporary attacks. This comprehensive evaluation under realistic scenarios underscores the importance of selecting appropriate algorithms for different types of IoT sensor attacks. The authors in [10], introduces a language for modelling false data injection attacks (FDIA) in IoT devices. The proposed approach supports processing data from all types of IoT devices using JSON format flattening techniques. It facilitates both simple and complex data alterations, enhancing the resilience of systems and improving the training of machine learning tools for attack detection. In [16] the authors present a survey on attacks targeting the interaction between cyber and physical worlds in cyber-physical systems (CPS). The paper categorizes and discusses signal injection and information leakage attacks, highlighting the need for a better understanding of these evolving threats and proposing new defense strategies.

The author in [1] proposes a conjugate gradient-based improved GAN (CG-IGAN) for anomaly detection in IoT networks. This approach leverages Independent Component Analysis (ICA) to extract features from a botnet dataset, which are then used to train a classifier. The CG-IGAN design improves the accuracy of detecting malicious IoT data, outperforming other state-of-the-art IoT-based anomaly detection methods. in [6] a machine learning algorithm to detect and mitigate false data injection attacks (FDIA) in wastewater treatment plants is presented. The study uses Linear Regression, K-Means clustering, and Auto Encoder neural networks to differentiate between clean and corrupted data, providing a robust anomaly detection and data cleaning solution for industrial IoT applications. In [7] authors focus on detecting False Data Injection (FDI) and Man-in-the-Middle (MitM) attacks in IoT and Industrial IoT (IIoT) environments. The proposed method achieves a 95% success rate in detecting these attacks by leveraging machine learning tools to analyse vulnerabilities in wireless communication protocols, emphasizing the critical need for security in smart systems. Authors in [4] develop a novel method using autoencoders (AEs) to detect false data injection (FDI) attacks in industrial IoT. By exploiting the correlation in sensor data across time and space, the proposed technique identifies and cleans falsified data, outperforming traditional support vector machine (SVM)-based approaches in both detection accuracy and data recovery. Authors in [5] propose an efficient prediction-based FDIA detection and location scheme (PDL) for smart grids. Utilizing vector autoregressive processes (VAR) to predict state vectors, the scheme detects and locates abnormal data by comparing predicted and observed measurements. The PDL approach simplifies the calibration process and demonstrates improved performance in extensive simulation results.

III. Preliminary and Problem

This section provides background of some terminologies and metrics used along with the problem in the paper for better understanding of our proposed approach.

A. Structural Similarity Index Measure (SSIM)

This metric assessed the perceptual similarity between manipulated and original frames. The metric quantifies how closely a processed or altred image resemble an origninal reference image while accounting for perpertual charateristics.

SSIM is calculated as follows.

$$SSIM(F_a, F_b) = \frac{(2\mu_a\mu_b+C_1)(2\sigma_{ab}+C_2)}{(\mu_a^2+\mu_b^2+C_1)(\sigma_a^2+\sigma_b^2+C_2)} \quad (1)$$

Where F_a, F_b are two consecutive video frames being compared. $\mu_a\mu_b$, the mean intensity values of frames F_a and F_b representing the average pixel brightness in each frame. σ_a^2, σ_b^2 , the variance of frames F_a and F_b quantifying the intensity variations within each frame. σ_{ab}, the covariance between frames F_a and F_b, measuring the degree of correlation in pixel intensity variations between the two frames. C_1 is the small constant to stabilize the equation when $\mu_a^2 + \mu_b^2$ is close to zero. C_2 is the small constant to prevent instability when $\sigma_a^2 + \sigma_b^2$ is close to zero.

B. Detection Rate

Detection rate, often termed as recall in classification tasks, is a metric that evaluates the percentage of true positive instances identified by a detection algorithm. In the context of video manipulation attacks, it quantifies how effectively an algorithm can identify manipulated frames within a video sequence. A higher detection rate signifies that the system is recognizing altered frames, even when the manipulations are subtle or dynamically applied. Detection rate is defined as:

$$DR = \frac{TP}{TP+FN} X100 \ (2)$$

Where TP: true Positives, the number of manipulated frames correctly identified as manipulated. FN: False Negatives, the number of manipulated frames incorrectly classified as normal.

C. Detection Accuracy

Detection accuracy is a measure of the overall correctness of a detection system, evaluating its ability to classify both manipulated and unmanipulated frames accurately. Unlike detection rate, which focuses solely on true positives, accuracy considers true negatives, making it a holistic performance indicator. Detection accuracy is computed as:

$$DA = \frac{TP+TN}{TP+TN+FP+FN} X \ 100 \ (3)$$

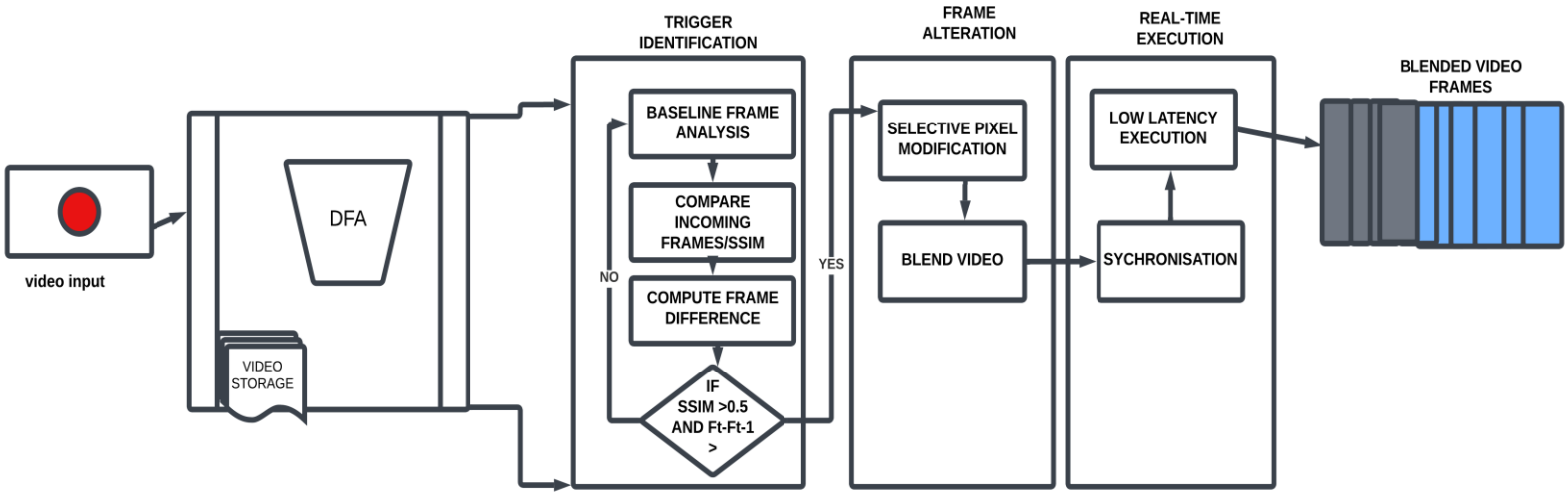
Where: *TP*: True Positive, *TN*: True Negatives, the number of normal frames correctly classified as normal, *FP*: False Positives, the number of normal frames incorrectly classified as manipulated, *FN*: False Negatives.

D. Problem Definition

The problem definition in this Paper is to achieve low detection rate and low detection accuracy with existing detection approaches when our proposed attack is applied that identifies attack instance based on SSIM in real Time

IV. DFA TECHNIQUE

The DFA attack dynamically alters video frames based on real-time contextual cues, designed to evade detection by maintaining perceptual continuity. Fig 1 shows a high-level overview of the proposed DFA method and its steps. Following are the details of each step.



A. Trigger Identification

Real-time monitoring identifies changes in lighting, motion, or scene stability using metrics like Structural Similarity Index Measure (SSIM). One of the most critical metrics in evaluating video manipulation attacks is the Structural Similarity Index Measure (SSIM), which quantifies the perceptual similarity between manipulated and original frames. A higher SSIM value indicates that the manipulation is less perceptible to the human eye, while a lower value suggests significant perceptual deviation.

The attack does not operate on a fixed schedule but instead analyses the variations in video streams to establish a dynamic manipulation threshold**.** This ensures that DFA remains stealthy by avoiding detection through abrupt scene changes. The dynamic threshold is calculated based on statistical variations in frame differences and structural consistency over time. To calculate the threshold for triggering DFA, the system first establishes a baseline of normal video behaviour. This is achieved by collecting an initial set of **50** consecutive frames and computing the frame difference and SSIM. Using this data, DFA calculates the threshold to maintain stealth and maximize the impact of the attack, DFA initiates frame manipulation only when low Frame difference and high SSIM conditions are met. When pixel variations between consecutive frames are at 15% of the average pixel intensity, it indicates **scene** stability. This provides the ideal condition for inserting manipulated frames without detection. If SSIM remains above 0.75**,** it suggests that the frames are perceptually similar. This allows DFA to introduce subtle alterations without raising suspicion.

B. Frame Alteration

Alterations are applied incrementally to avoid perceptual disruption. The alterations are applied over a sequence of frames to smooth transitions and maintain perceptual consistency. First the difference between consecutive video frames is calculated. This helps identify the amount of change that has occurred between frames. Let F_t represent the frame at time t. The frame difference D_t is calculated as:

$$D_t = |F_t - F_{t-1}| \quad (4)$$

D_t represents the frame difference at time *t*, F_t is the current video frame at time *t*, and F_{t-1} is the previous frame at time *t*, the absolute difference between corresponding pixel values is calculated to determine the extent of variation.

The blending coefficient is adjusted dynamically, adjusted based on scene complexity, if the scene remains static with minor pixel variations, a higher blending ratio is applied. If there is motion or object displacement, a lower blending ratio is used to maintain consistency. Changes include pixel-level adjustments and blending with pre-recorded video which is performed in a systematic manner to ensure seamless integration of manipulated content with real-time video frames. This step is executed using a combination of pixel interpolation, weighted averaging, and spatial-temporal consistency to avoid abrupt transitions that could trigger detection mechanisms. A pre-recorded frame is retrieved based on scene context to maintain continuity. The pre-recorded frame is blended with the real-time frame using a weighted combination function:

$$F_t' = (1 - \alpha)F_t + \alpha P_t \quad (5)$$

Where P_t is the pre-recorded frame and α controls the blending ratio, dynamically adjusted to maintain perceptual continuity.

C. Real Time manipulation

Manipulation is performed in real time, each incoming frame Ft is analyzed for key visual properties such as brightness, contrast, and edge sharpness. A mapping function is applied to modify the pixel intensities based on a transformation mode. Instead of abruptly replacing frames, DFA performs weighted averaging between real-time frames and pre-recorded sequences maintaining operational integrity and stealth. The

real-time execution of DFA is triggered based on the frame difference calculation and structural similarity index (SSIM) thresholds. The manipulation begins when the frame difference D_t and SSIM drop below dynamic thresholds:

$$Trigger = \begin{cases} 1, \\ 0, \end{cases} if\, D_t and\, SSIM < \gamma \quad (6)$$

Where $\sigma$and γ are empirically determined thresholds.

DFA actively monitors incoming frames to determine the optimal moment for initiating alterations. This ensures that the attack is not only more stealthy but also adaptive to real world conditions such as smart home systems, intelligent vehicle systems and disaster management settings. DFA's execution pipeline which involves capturing and analyzing video frames, subtle frame modifications using pixel-level adjustment and data blending, is designed to perform these in-frame modifications using OpenCV's efficient image processing functions cv2.absdiff() to compute absolute frame differences to detect subtle scene changes, cv2.GaussianBlur() to smoothen manipulated frames for seamless blending, cv2.addweighted() to merge real-time and pre-recorded frames to maintain visual consistency, ensuring that each frame is processed within a minimal time window. This prevents noticeable lag or latency spikes that could expose the attack.

V. EXPERIMENTAL RESULTS

We discuss the comprehensive experimental framework employed to implement and evaluate the proposed Dynamic Frame Alteration (DFA) attack. The experimental framework is designed to replicate a realistic IoT video surveillance environment and assess the impact of DFA and other traditional attacks under controlled conditions. Python and OpenCV were instrumental in executing and analyzing the experiments, providing robust tools for real-time video manipulation and processing.

A. Hardware Configuration

The core processing unit was a Raspberry Pi 3 Model B board with 4GB RAM and a quad-core ARM Cortex-A72 CPU. This device served as the computational hub for processing video feeds, implementing attacks, and running detection algorithms. A Camera Module V2 with an 8-megapixel sensor provided real-time video feeds. The camera was mounted in a stable environment to capture consistent frames for experimentation. An HDMI monitor connected to the embedded board displayed the manipulated video feed and detection results. A 32GB microSD card stored experimental logs, video frames, and algorithm outputs.

B. Operating System

Raspbian OS (based on Debian Linux), used for video capture, frame manipulation, and structural similarity index calculation. Scikit-learn for implemented machine learning models for attack detection. NumPy supported numerical operations. A real-time video feed was captured using OpenCV's cv2.VideoCapture function, which initialized the camera module and provided continuous frame streams. The frames were resized to 640x480 pixels for consistent processing.

C. Comparative Attacks

Three relevant and popular attacks are considered and implemented for comparison.

Replay Attack: Pre-recorded video sequences replace the live feed, maintaining high SSIM values but introducing temporal inconsistencies. Earlier frames are re-sent at a later time, replacing the live feed at the chosen moment. [22]

Frame Injection: New or manipulated frames overwrite or insert themselves at specific intervals, while the rest of the feed remains genuine. Extraneous frames were injected periodically using OpenCV's cv2.imshow function. [4]

Video Stream Hijacking: From a certain point onward, all frames are replaced entirely with an attacker-controlled feed. [23]

D. Detection Algorithms Considered

The effectiveness of DFA and traditional attacks was evaluated using three popular detection algorithms.

Support Vector Machine (SVM**):** SVM models classify frames based on features like SSIM and temporal continuity.[14]

k-Nearest Neighbours (kNN): Anomalies are identified by comparing frames to a reference set using Euclidean distance.[5]

Local Outlier Factor (LOF): LOF detects outliers based on local density deviation. [18]

A. SSIM Comparison

Table 1 below outlines SSIM results of various attacks while varying detection approaches are applied.

TABLE 1. SSIM ANALYSIS ACROSS ATTACK AND DETECTION METHODS

Detection Approach	Attack Type			
	Replay	Frame Injection	Video Stream Hijack	DFA
LOF	0.735	0.404	0.735	0.890
SVM	0.758	0.411	0.758	0.898
kNN	0.745	0.379	0.395	0.908
Average	0.746	0.398	0.400	0.898

The DFA attack showed the least perceptual deviation among all attack types. The SSIM values for DFA remained relatively high averaging around 0.85. This result is attributed to the dynamic nature of the attack, where alterations were applied incrementally and contextually, blending them seamlessly with the original frames to maintain perceptual continuity. The algorithm's real-time analysis of changes in lighting, motion, and scene stability allowed it to adjust the alterations to be subtle and unnoticeable to the human eye. The SSIM values never dropped below 0.80, indicating that the attack remained stealthy and difficult to detect visually.
The SSIM values for replay attacks were consistently high

(around 0.95), as the pre-recorded video sequences were identical to the original video, meaning no perceptual disruption occurred. However, the high SSIM values masked the temporal inconsistencies introduced by the attack. Replay attacks were easy to detect when temporal analysis, such as motion patterns, was employed. Frame injection attacks had lower SSIM values compared to DFA and replay attacks. The abrupt inclusion of extraneous frames into the video feed disrupted its continuity, leading to perceptible anomalies. The SSIM values dropped significantly, with averages around 0.50–0.30, depending on the number of frames injected. This lower SSIM reflected the noticeable disruption caused by the injection, making it more detectable in comparison to DFA.

Video stream hijacking produced the most significant visual discrepancies, leading to very low SSIM values (around 0.40–0.30). This attack replaces the entire video stream, resulting in drastic differences between the original and manipulated frames. The lower SSIM values indicate that this attack was easily detectable through visual inspection, making it less stealthy than DFA. SSIM evaluates the perceptual similarity between manipulated and original frames. High SSIM values suggest less perceptible alterations, making the manipulation more effective.

DFA maintains the highest SSIM values (average: 0.898381) across detection algorithms, indicating that its manipulations are subtle and blend seamlessly into the video feed. The incremental alteration ensures the manipulation is imperceptible while maintaining temporal and spatial consistency. The SSIM values for Frame Injection (average:0.398148) are significantly lower, reflecting abrupt and perceptually jarring anomalies caused by inserting extraneous frames. Replay attack s achieve moderately high SSIM values (average: 0.746005). The lack of real-time contextual relevance results in detectable temporal inconsistencies despite high similarity. Video Stream Hijacking shows SSIM values (average: 0.399985), as the complete replacement of the video feed introduces significant perceptual deviations.

B. Detection Rate Comparison

Table 2 summarizes detection rates across attacks and detection algorithm

TABLE 2. DETECTION RATE ANALYSIS ACROSS ATTACK AND DETECTION METHODS

Detection Approach	Attack Type			
	Replay	Frame Injection	Video Stream Hijack	DFA
LOF	87.20%	64.59%	75.60%	10.33%
SVM	87.01%	61.05%	72.75%	9.14%
kNN	86.96%	59.97%	71.99%	10.89%
Average	87.05%	61.87%	73.45%	10.12%

Replay attacks had the highest detection rate, averaging 87.05%. This is because replayed frames often lack temporal coherence with live video, making them easier for algorithms to detect. Both kNN and SVM performed exceptionally well, leveraging temporal patterns and feature space continuity to identify repeated sequences. These high detection rates were a result of the algorithms' ability to identify the mismatch between the original and replayed video frames, especially when the video feed exhibited unnatural temporal patterns or motion. Frame injection attacks yielded a detection rate of **61.87%** on average. The inconsistency introduced by injecting extraneous frames disrupts the temporal flow, providing detectable anomalies. However, the detection rate varied significantly between algorithms due to differing sensitivity to temporal disruptions.

kNN and LOF performed exceptionally well in identifying the injected frames, as these algorithms excel in detecting temporal and spatial inconsistencies. Video stream hijacking had the highest detection rate among all attacks, this attack recorded a moderate detection rate of 73.44%, as it replaces the entire video stream, introducing noticeable deviations in both spatial and temporal features. All three detection algorithms were highly effective in detecting hijacking, with SVM achieving the highest detection rate (99%). Algorithms like LOF and kNN demonstrated difficulty in detecting DFA due to its dynamic nature. LOF's reliance on local density deviations fails when manipulations remain within acceptable variance ranges. Similarly, kNN's distance-based comparison struggles with gradual alterations that remain close to the reference set. Detection rates varied with threshold adjustments in each algorithm. For instance, lowering the SSIM threshold in SVM improved recall but also increased false positives.

C. Detection Accuracy

Table 3 below summarizes detection accuracy across attacks and detection algorithms.

TABLE 3. DETECTION ACCURACY ANALYSIS ACROSS ATTACK AND DETECTION METHODS

Detection Approach	Attack Type			
	Replay	Frame Injection	Video Stream Hijack	DFA
LOF	69.68%	69.08%	69.30%	20.76%
SVM	90.90%	73.39%	69.97%	20.46%
kNN	90.50%	70.64%	69.46%	20.37%
Average	83.69%	70.70%	69.59%	20.53%

DFA's detection accuracy averaged 20.52%, reflecting the algorithm's inability to differentiate manipulated frames from normal ones. This low accuracy underscores DFA's capability to evade detection through dynamic and context-aware manipulations. Both kNN and LOF exhibited poor performance, highlighting the need for enhanced feature extraction techniques. Replay attacks achieved an average accuracy of 83.69%, with SVM outperforming other models. The high temporal similarity of replay attacks simplifies the classification task, enhancing true positive identification while reducing false negatives. Frame injection attacks recorded a moderate accuracy of 70.70%. Despite the abrupt anomalies introduced by extraneous frames, false positives were more

common, as algorithms occasionally misclassified normal frames with similar patterns. Video stream hijacking achieved a detection accuracy of 69.57%, largely due to the significant visual deviations introduced.

However, the complete replacement of the video stream occasionally led to false positives in dense anomaly detection algorithms like LOF. DFA manipulation often remained undetected due to its subtlety, leading to higher false negatives and lower accuracy. Replay attacks, on the other hand, maintained high accuracy but suffered from occasional false positives in algorithms relying on temporal consistency. SVM outperformed LOF and kNN across most attack scenarios due to its robust feature extraction and decision boundary optimization.

D. False Positives and False Negatives

Due to its stealthy nature, the DFA attack resulted in a moderate number of false negatives (approximately 20–30%). However, false positives were relatively low, with values around 10%. This indicates that while DFA was challenging to detect, the algorithms performed reasonably well in avoiding incorrect classifications. False positives for replay attacks were minimal, as the detection algorithms could easily distinguish replayed frames from the live feed. However, false negatives were rare, occurring in less than 5% of the cases. This high detection reliability was expected due to the simplicity of the replay attack and the ability of the algorithms to detect temporal inconsistencies. False positives for frame injection attacks were slightly higher than for replay attacks, as the injection of multiple frames could occasionally be mistaken for normal video fluctuations, especially when the injected frames were similar to the original feed. False negatives were rare but not non-existent, with rates around 5–10%. This reflects the effectiveness of detection algorithms in identifying abrupt changes in the video sequence. False positives and false negatives for video stream hijacking were minimal due to the significant visual discrepancies between the hijacked and original video feeds. The algorithms were highly reliable in detecting this attack, with false positives and negatives both close to 0%. The computational overhead and time taken per frame are important factors when considering the feasibility of real-time detection in video surveillance systems.

V. CONCLUSION

The findings in this research underline the critical vulnerabilities present in IoT-based video surveillance systems when subjected to the novel Dynamic Frame Alteration (DFA) attack. Unlike conventional attacks such as replay attacks, frame injection, and video stream hijacking, DFA introduces a unique and sophisticated approach to video manipulation that significantly challenges existing detection mechanisms. These scenarios highlight the urgent need for system designers to incorporate multi-layered security mechanisms, such as redundancy in sensor inputs, blockchain for data integrity, and real-time cross-validation of video and sensor data. Future work will focus designing detection and defense mechanisms to counter DFA attack.

VI. REFERENCES

[1]. R. Manivannan, "Improving IoT Security with AI-Powered Anomaly Detection and Intrusion Prevention," *2023 (ICSES)*

[2]. A. Font, J. Jarauta, R. Gesteira, R. Palacios and G. López, "Threat models for vulnerability analysis of IoT devices for Manipulation of Demand attacks," *2023 (JNIC)*.

[3]. H. Jahangir, S. Lakshminarayana, C. Maple and G. Epiphaniou, "A Deep-Learning-Based Solution for Securing the Power Grid Against Load Altering Threats by IoT-Enabled Devices," 2023, in *IEEE Internet of Things Journal*

[4]. M. M. N. Aboelwafa, K. G. Seddik, M. H. Eldefrawy, Y. Gadallah and M. Gidlund, "A Machine-Learning-Based Technique for False Data Injection Attacks Detection in Industrial IoT,"2020 in *IEEE Internet of Things Journal*.

[5]. Shi, W., Wang, Y., Jin, Q., & Ma, J. (2018). PDL: An Efficient Prediction-Based False Data Injection Attack Detection and Location in Smart Grid. *2018 (COMPSAC)*

[6]. S. Gönen, M. A. Barişkan, D. Y. Kaplan, E. N. Yilmaz and A. Çetin, "A Novel Approach Detection for False Data Injection, and Man in the Middle Attacks in IoT and IIoT," *2023 IEEE PES GTD.*

[7]. A. Parvathy, G. Leela Kasyap, D. Venkata Abhinav, A. N. V. Surya Sai, R. Sriranjani and N. Hemavathi, "Hybrid Machine Learning based False Data Injection Attack Detection and Mitigation Model for Wastewater Treatment Plant," *2022 (ICACRS)*

[8]. W. Zeng, H. Hu, Q. Guo and D. Zhou, "A Mimic Cloud Ruling Method for Defending Against Time-Delayed Covert Channel Attacks," *2021 (ISPDS)*

[9]. P. Chaudhary, B. B. Gupta, K. T. Chui and S. Yamaguchi, "Shielding Smart Home IoT Devices against Adverse Effects of XSS using AI model," *2021 IEEE International Conference on Consumer Electronics (ICCE)*

[10]. M. Briland and F. Bouquet, "A Language for Modelling False Data Injection Attacks in Internet of Things," (*SERP4IoT),*

[11]. J. Phukan, K. F. Li and F. Gebali, "Hardware Covert Attacks and Countermeasures,"*2016 (AINA)*.

[12]. Pu, C. Spam DIS attack against routing protocol in the Internet of Things. *Wireless Personal Communications, 2019 (CNC)*

[13]. M. K. Verma and R. K. Dwivedi, "A Survey on Wormhole Attack Detection and Prevention Techniques in Wireless Sensor Networks," *2020 (ICE3)*

[14]. S. K, S. V, A. Singh, A. R, H. Saxena and S. S. S, "Detection and Mitigation of Man-in-the-Middle Attack in IoT through Alternate Routing," *2022 (ICCMC)*

[15]. I. U. Khan, M. Y. Ayub, A. Abdollahi and A. Dutta, "A Hybrid Deep Learning Model-Based Intrusion Detection System for Emergency Planning Using IoT-Network," (*ICT-DM)*

[16]. A. Mosenia, S. Sur-Kolay, A. Raghunathan and N. K. Jha, "DISASTER: Dedicated Intelligent Security Attacks on Sensor-Triggered Emergency Responses," in 2017 IEEE Transactions on Multi-Scale Computing Systems

[17]. A. Tandon and P. Srivastava, "Trust-based Enhanced Secure Routing against Rank and Sybil Attacks in IoT," 2019 (IC3)

[18]. J. Lorandel, M. A. Khelif and O. Romain, "A Low-cost Hardware Attack Detection Solution for IoT Devices," 2022 IEEE 31st International Symposium on Industrial Electronics (ISIE)

[19]. A. Alaali and W. Elmedany, "Hardware Trojan and Countermeasures for Internet of Things," 2022 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)

[20]. T. Mizuno, H. Nishikawa, X. Kong and H. Tomiyama, "Empirical Analysis of Side-Channel Attack Resistance of HLS-designed AES Circuits," 2023 (ICEIC)

[21]. M. E. Deowan, S. Haque, J. Islam, M. Hanjalayeamin, M. T. Islam and R. Tabassum Meghla, "Smart Early Flood Monitoring System Using IoT," *2022 14th*

[22]. A. A. Elsaeidy, A. Jamalipour and K. S. Munasinghe, "A Hybrid Deep Learning Approach for Replay and DDoS Attack Detection in a Smart City," 2021 in *IEEE Access*, vol. 9

[23]. H. Kooshkaki, B. Akbari and A. Ghaffari Sheshjavani, "A Multi-Level reputation-based pollution attacks detection and prevention in P2P streaming," *2016 (IST)*