UNIVERSITY OF AMSTERDAM

RP1

Detection of false frame attacks in video systems

February 7, 2021

Student:
Joris Shuko Janssen
11248823

Tutor:
Prof. Z.J.M.H. Geradts

Course:
Network & Security Engineering

Abstract

Camera systems are very important in the current security landscape, where a minimal amount of personnel is required to monitor a large or remote area. In this research project the viability of multiple detection methods are discussed in relation to a frame duplication attack. These detection methods should be possible to apply to any generic camera system and ideally require little to no adjustments to the camera or its surroundings. The first method is movement based detection in which the server will task a camera to perform a specific movement and verify if it actually performs this movement. A second detection option looks at the feasibility of recording the Electrical Net Frequency(ENF) through the microphone to verify whether the audio is forged or not. A big advantage of using the ENF recording of a camera is that it is essentially a random number generator that is verifiable by correlating the measurement of the ENF at both the camera and the central server.

Contents

1 Introduction

In the last few decades video security systems have become more popular[14, 11]. Companies use camera systems more and more to monitor remote locations, both outside and inside buildings. This popularity is also seen among private citizens as video systems in the form of Ring doorbell or Google nest camera have become common. This rise in popularity, combined with a more general adaptation of IOT devices has increased the diversity of different hardware, software, and protocol implementations. However this popularity also has a darker side, hackers will more frequently try and target these camera systems to either gain access to the camera feed or disable it entirely. The scope of this research project is that an attacker has gained access to a camera and tries to use it to hide himself from the feed. In this project the attacker is using a frame duplication attack. This attack consists of an attacker that will send prerecorded frames to the server instead of a live feed. For companies that employ a large amount of cameras the setup is as follows: the cameras are connected to a single server which shows the feeds to security personnel (figure 1). Sometimes, this server will run software performing motion detection to notify guards if motion or noise is detected in the view of the camera. The detection methods will be ran on these servers. In order to allow for easy adaptation we have set the requirement that there should be as little adjustments to the camera as possible. For each detection method we will have an in-depth explanation of how it works, if it is usable in the current landscape, and if it is possible for an attacker to defend him/herself against it.

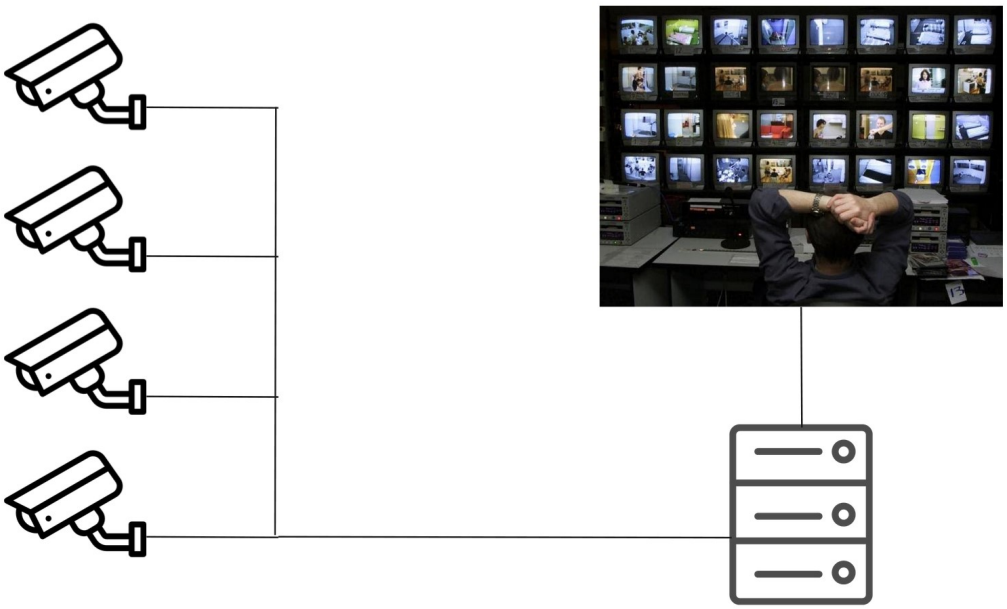

Figure 1: Example setup of a camera system.

2 Background

2.1 Motion Detection

Motion detection is commonly applied in security systems. Most motion detection uses background removal to detect movement, as anything not part of the background is motion. This background detection can be done in two ways, static or dynamic. A static background remover works by taking a single image of the background. This image is then subtracted

from every frame. Any differences are then classified as movement. The static method is quick, but if the camera moves a little bit or the light changes, the entire image will be classified as movement. A more robust alternative to static background detection is a dynamic background model in which the definition of the background changes depending on what is in view. This means that if something is standing still for long enough it is most likely part of the background or a light change. In this project the OpenCV 'createBackgroundSubtractorKNN'[10] function is used to remove the background. This is a dynamic algorithm. An example can be seen in figure 2. A video is fed frame by frame to the background subtractor and the result can be seen in figure 3. As can be seen in the image, the moving humans are detected, as well as their shadows. Also noteworthy is that there are small spots in the background as well as the red-white warning line that are seen as movement. A subsequent contour finding algorithm should distinguish between small artifacts and actual movement.

Figure 2: Frame

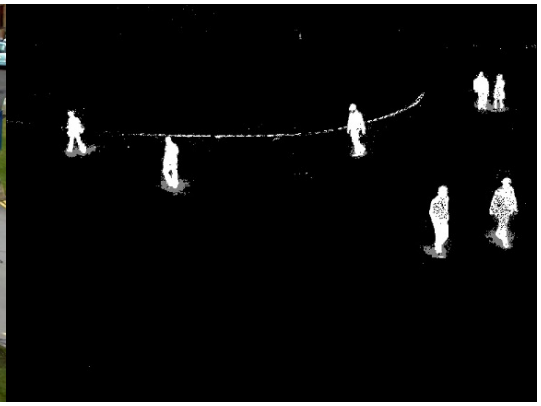

Figure 3: Background and shadows

2.2 Features

Features of an image are any type of local information about that image. An example of this is Scale Invariant Feature Transform (SIFT)[5]. With SIFT a feature is represented by a histogram of oriented gradients. The major advantage is that this algorithm is scale invariant. This means that if an image is zoomed in, the same features are still found with the same values. However SIFT is quite computationally expensive. While there are faster alternatives such as ORB[12] or SURF[1], neither is advised to run on every frame of multiple cameras. The performance of feature extraction depends on the size of the picture. If performance becomes a problem, the image can be scaled down to trade accuracy for performance. An example of these features can be seen in figure 4, where every circle describes a feature.

Figure 4: SIFT example

2.3 Transformation Matrix

A transformation matrix is a matrix that maps coordinates of an image to new coordinates. In the example of figure 5 and 6 an affine transformation is applied to rotate and shear the image. This matrix is a composite matrix of all the separate operations. This means that all transformations can be calculated separately and multiplied with each other to create a single transformation matrix to apply to an image.

Figure 5: Normal image

Figure 6: Transformed image

Figure 33 in the appendix shows all the separate transformations possible with a transformation matrix.

2.4 FFT

The Fast Fourier Transform[9] is an algorithm used to break down a signal into its base frequencies. Subsequently this allows us to determine the presence of a specific frequency

in the original signal. In figure 7 a signal can be seen that breaks down into two signals of approximately 40Hz and 90Hz (figure 8).

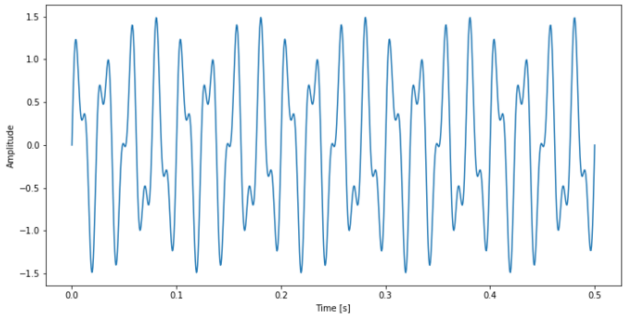

Figure 7: Signal

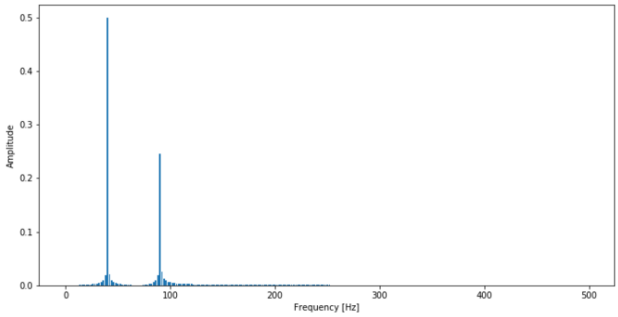

Figure 8: FFT result

3 Attack model

In modern systems cameras have been becoming increasingly powerful, capable of running complex image operations on the camera module itself. An example of such a complex operation is facial recognition that can easily run on a ESP32 microprocessor [2]. An attacker can utilize these additional processing capabilities to more easily hide him/herself. The model used in this project can be seen in figure 9. If no movement or noise is detected, the frames will be recorded and then as soon as a trigger is detected the prerecorded frames will be sent to the server. A trigger can be anything, from motion being detected or the face of the attacker being seen in the frame.

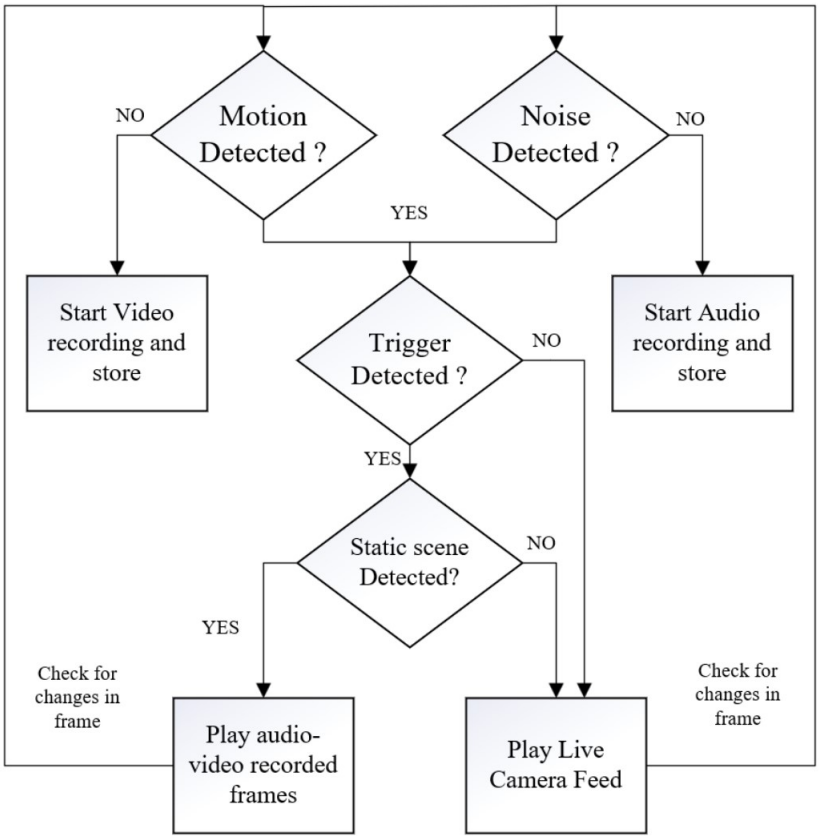

Figure 9: Attack model[8]

As can be seen in figure 10 the algorithm will show a live feed when no motion is detected. As soon as any kind of motion is detected the prerecorded frames will be sent out to the server, thereby hiding any movement in the original image. This specific method will also work in a hallway where movement is common, by triggering on seeing the attackers face and then sending prerecorded frames.

Figure 10: Top: Frames being send by camera, Bottom: Real frames

3.1 Real Life Examples

So far no attacks utilizing this model or similar ones, have been observed in practice. On the other hand, attacks on camera systems are more frequent. These attacks mainly focus

on IOT devices that are usually a lot cheaper and less well secured then enterprise camera systems.

4 Protection Methods

In this section we will discuss three methods of detecting these false frame attacks. Each of these methods works with existing camera systems and doesn't require altering existing camera installations. For each method the viability, upsides, and downsides will be discussed. In the next section specific implementations will be shown.

4.1 Hashing based detection

In software engineering, using a hashmap is a common method to detect duplicate data. For each input the hash will be calculated and compared to previously stored hashes, if an entry already exists it is a duplicate. This is however not directly applicable to images as two similar images will produce different hashes. A single pixel being a different colour will produce a completely different hash, this can be seen in figure 11 and 12. For our purposes, this poses a problem. We wish to have two similar images produce a similar or identical hash in order to compare these images, which a normal hashing algorithm does not provide.

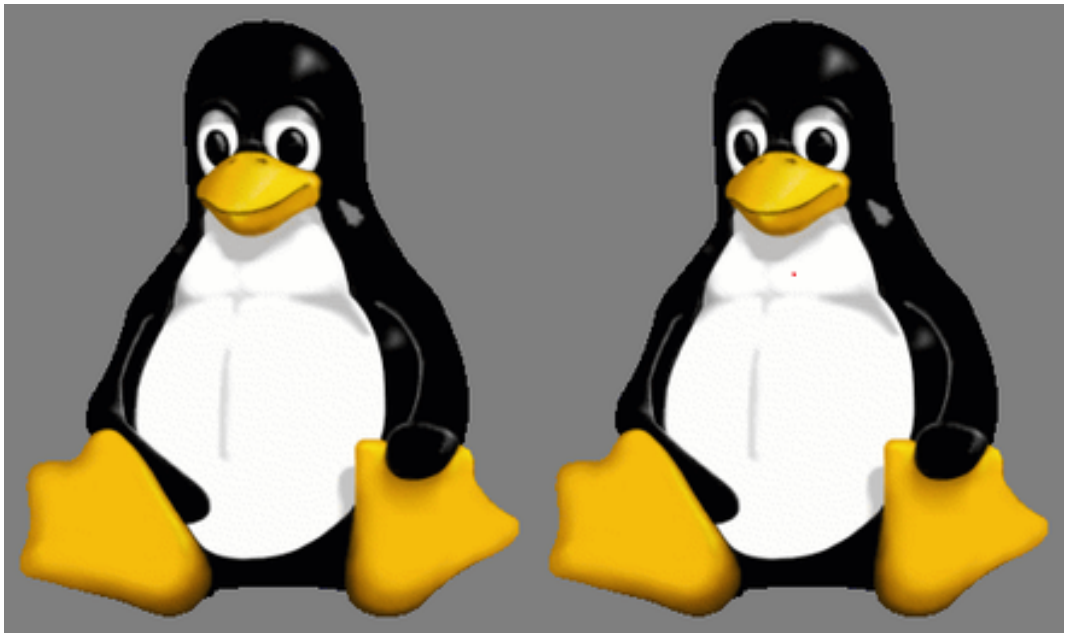

Figure 11: Normal image
MD5 Hash: 19...0c

Figure 12: One pixel changed to red:
MD5 Hash: 6a...54

A second approach would be to use locality sensitive hashing where a neural net is used to give images with the same content the same hash[4]. This method essentially labels images based on their content. An empty hallway would get an "Empty" label while a hallway with a single person wearing a red shirt would get a "Red shirt" Label.

4.1.1 Difficulty of detection

While hashing is a relatively simple operation capable of being applied to every frame, it does not prove anything. If two frames are the same it does not mean that anything malicious is happening, it simply means the camera is still seeing the same thing. It might be possible to

use locality sensitive hashing to create a timeline but that still would not prove whether or not a false frame attack is currently being employed. A timeline throughout a building would track the movements of every individual by giving them labels. This is however something that requires a lot of manual work as multiple people wearing red shirts would be seen as the same person by a simpler system. And a more advanced system that uses for example facial recognition to construct a timeline would take more processing power, which makes it infeasible to use in practice.

4.1.2 Attacker protection

If a system uses normal hashes to determine duplicate frames an attacker can simply add random values to frames in such a way that they are not visible to the user (the colours #141414 and #1c1c1c look pretty much identical to humans) as well as calculate hashes themselves to prevent collisions.

4.2 Motion based detection

In cryptography a common problem is that a system has to proof it has access to a hidden key, without actually showing this key. A cryptographic challenge is then given too verify possession of a private key. A similar proof challenge can be implemented in camera systems, where a server tasks a camera with making a movement. The server then checks whether the movement actually is made by using feature matching to create a transformation matrix. This requires a camera that has the option to either move, rotate, or zoom.

To obtain the direction of movement, a translation matrix is calculated. The first step to calculate this matrix is to find features in two images as seen in figure 14 and 15. The same features found in one image are also visible in the other image. The next step is to match the features of one image with the features on the second image, this can be done by brute force. There are however more efficient algorithm to match features, such as FLANN[6]. The matches found can be seen in figure 16, these are ten matched features and as can be seen the features are matched correctly between both images. The two sets of matches represent two sets of coordinates, from which a translation matrix can be created, in the case of our example image the matrix can is given in figure 13. The values are rounded up for easy readability.

$$\begin{bmatrix} 1 & 0 & 208 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Figure 13: Translation matrix created from matches in figure 16

The value to note is the 208, which means that the image has been moved 208 pixels to the right (see figure 33), this implies that the camera moved to the left. Using this method it is possible to determine zoom or movement from two frames.

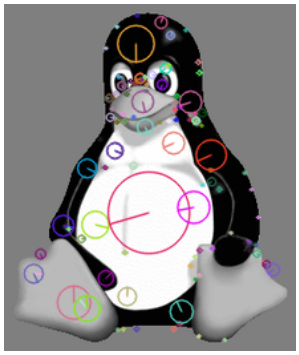

Figure 14: Features in normal image

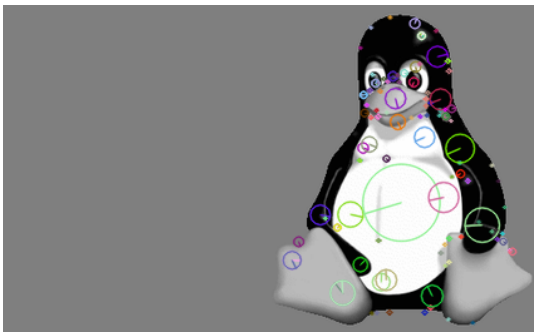

Figure 15: Features in image where camera was moved to the left

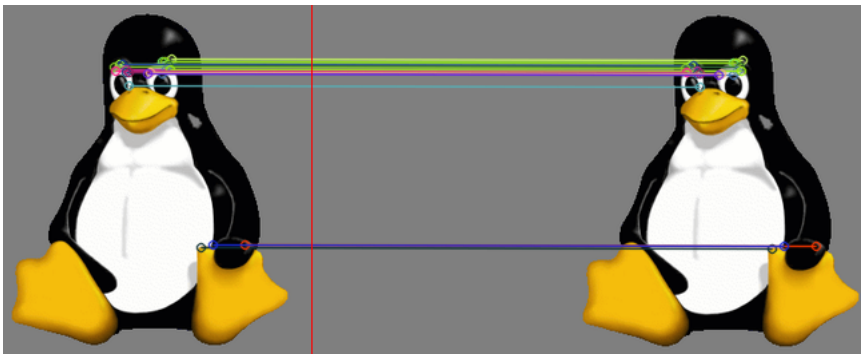

Figure 16: Matched features between two images, red line to distinguish the two images.

4.2.1 Difficulty of detection

After performing feature matching and calculating the transformation matrix we can verify whether the camera has performed the tasked movement. However there are still two notable problems with this method. Firstly, feature matching and extraction are expensive algorithms and can therefore not run every single frame if a large amount of cameras are installed in the system. In this scenario the algorithm would have to run periodically, looking at frame n and $n+x$ instead. Apart from putting less load on the system it also gives two clearer frames as the time in between gives a camera time to focus after moving.
Secondly, this algorithm requires a movable camera, which not all cameras are capable of. This algorithm does not allow an investigator to check afterwards whether frames are forged or not.

4.2.2 Attacker protection

It is possible for an attacker to evade movement based detection by stitching images together. If an attacker has recorded an area (denoted in red, figure 17) and needs to move to the top left (denoted in green, figure 18). It is possible to create a composite frame as seen in figure 19.

Figure 17: Red area is recorded area

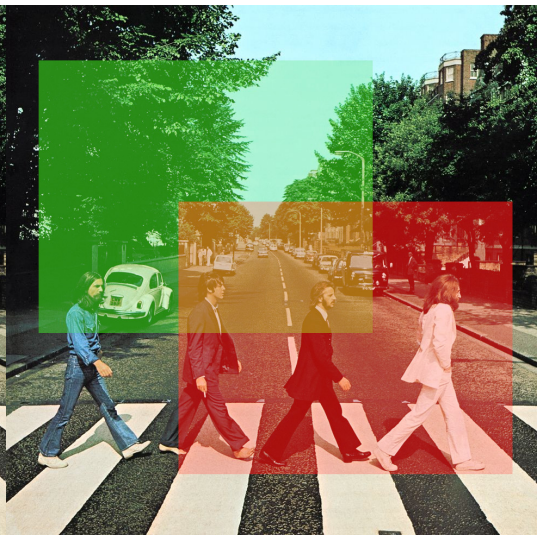

Figure 18: Green area is area to move to

Figure 19: composite image

A second option is to constantly move the camera along the complete image and create a composite frame consisting of prerecorded images.
However both evasion options create problems that can be detected both by humans as well as by computers. The first problem is that stitching is not perfect and can create a seam (figure 20). These seams are detectable by humans, or by looking for lines using ridge detection.

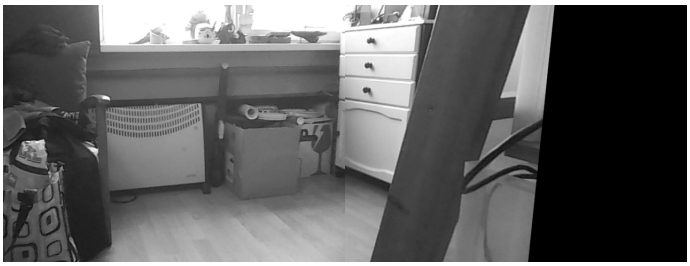

Figure 20: Seam created by stitching

A second problem is that modern cameras will adapt to their surrounding light levels

using auto white balance. If a camera points at a bright area the rest will darken and vice versa. This means that if a bright image and a dark image are stitched together it creates the effect visible in figure 21.

Figure 21: Light/Dark contrast created by stitching

4.3 ENF based detection

The European power grid operates at an AC voltage of 230 volt. this means the potential of the live wire alternates between -115 and 115 volt multiple times per second. In Europe this frequency is 50Hz(±10mHz) and in the US the frequency is 60Hz (±20mHz). The Electrical Net Frequency (ENF) has the same frequency across different nearby buildings. As can be seen in figure 22, the deviation of the ENF has a high correlation when measured at different buildings.

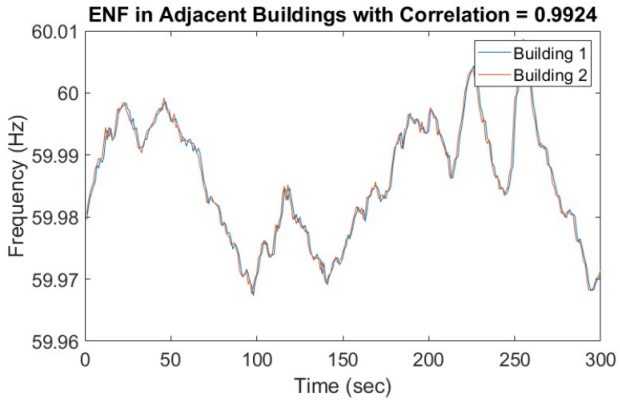

Figure 4. ENF captured in adjacent Buildings.

Figure 22: Correlation between ENF measured at different buildings[8]

This means that a camera could records the ENF and send it together with the video to the server. The server can then verify whether or not the video is live or forged. However this would require adding a oscilloscope to every camera which would also allow an attacker to simply take the output of the oscilloscope and send it with a forged video. Therefore the recording needs to be integrated into either the video or the audio. Fortunately electrical devices make a sound based on the ENF. The attack using audio works the same way as with video (figure 23) where if no noise is detected the silence is recorded and send whenever any noise is detected.

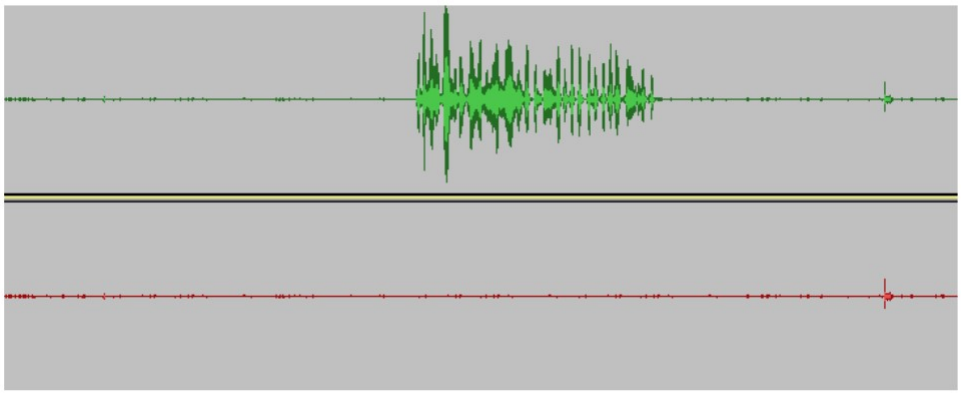

Figure 23: Audio Duplication attack example. Green is real audio. Red is being send by attacker[8]

Since the authors performed this research in the US, the ENF is 60Hz. The first step is to create a spectrogram of the audio received from the camera (figure 24). In this figure three horizontal lines are clearly visible at 60, 120 and 180 Hz. These are the odd harmonics of the 60 Hz sound coming from the audio source (A laptop in this case)

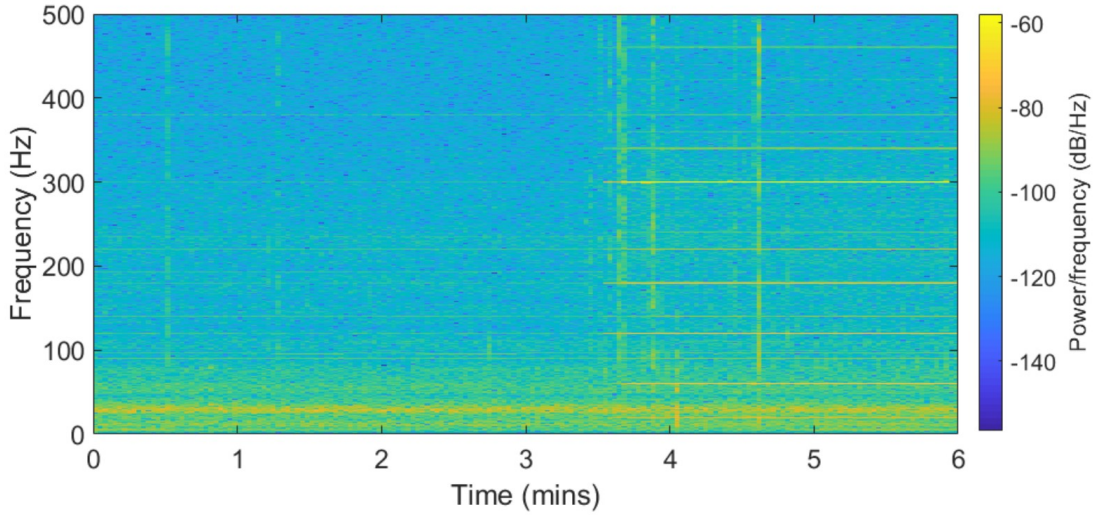

Figure 2. Spectrogram of audio recording with noise source after 3.5 min.

Figure 24: [8]

Now FFT can be used to extract the separate 60, 180, 300Hz signals (figure 25. The 180 and 300 Hz signals show similar ENF fluctuations.

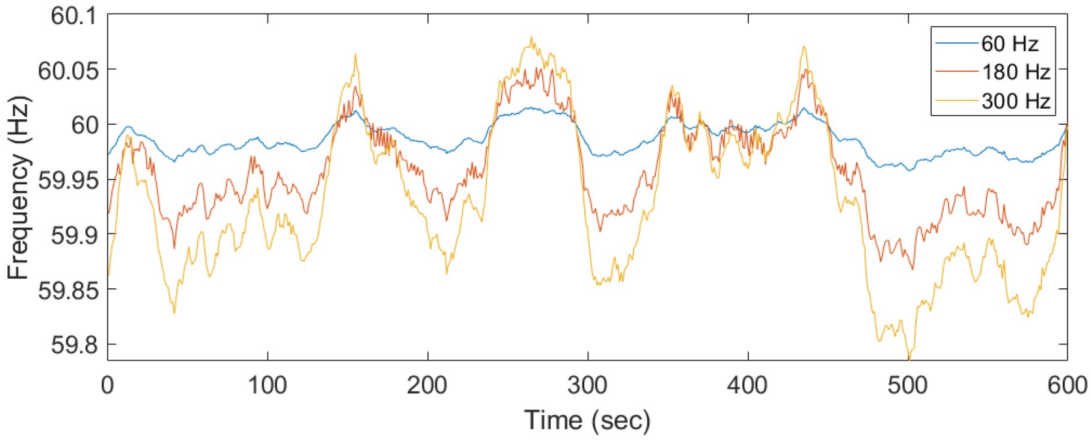

Figure 8. Different harmonics of power recording shifted to 60 Hz for comparison.

Figure 25: [8]

This ENF recording can now be correlated with data from an oscilloscope (or other sources for ENF) at the server. If the correlation drops too low it can mean that an attacker is sending a forged signal.

4.3.1 Application of detection

Doing false frame attack detection based on ENF has the major advantage that FFT is a very fast algorithm, which means that it is not an issue if the algorithm is run on all cameras at the same time. But this is not necessary as it is also possible to retroactively check whether or not a given audio stream was spoofed. The only thing that a sever needs to do is record the ENF at the server. A second advantage is that it is possible to run this detection without any modifications to the cameras themselves. It does however require a microphone to be present on a camera and a source for the noise to be recorded.

4.3.2 Attacker protection

It is more difficult for an attacker to protect him/herself against this detection method than with the other detection methods. An option would be to extract the 60/180/300 Hz signal from the real signal and insert this into the forged audio stream. More research is needed to verify if this is a valid option. It is possible for an attacker to try and predict the ENF signal, but this is difficult to do[3].

5 Results

The experiments where performed using a logitech C270 webcam connected to a PC. The code for the results is written in Python to allow access to the powerful OpenCV library [10] as well as for easy programming.

5.1 Motion based

For motion based detection the algorithm was applied whenever a button was pressed. During this process we found the following problems:

- If the camera is still moving when the second frame is taken the detection become unreliable (figure 26):

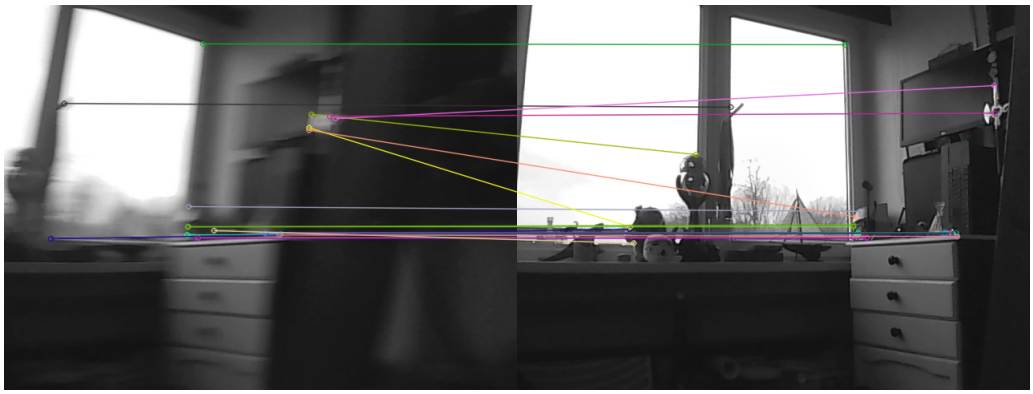

Figure 26: Invalid matches due to movement

- If there is a light intensity change, the matches also are incorrect sometimes (figure 27):

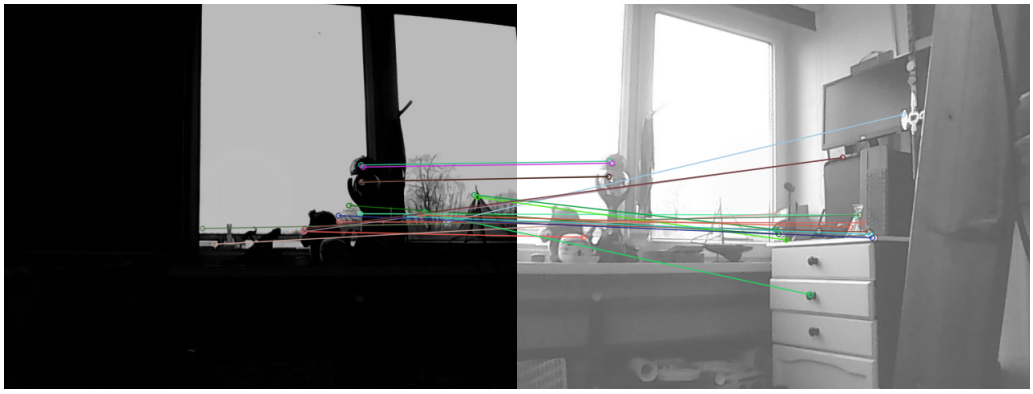

Figure 27: Invalid matches due to light (exaggerated example)

- If the camera moves too much such that part of the image is out of view, it also cannot find correct matches (figure 28):

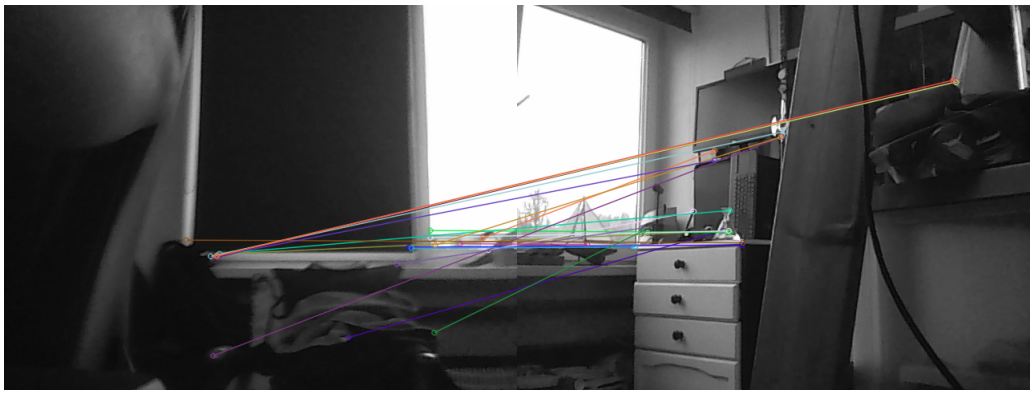

Figure 28: Invalid matches due to to much movement

Due to these problems a single missed challenge should not result in an alarm going off,

but could be indication for a human operator to perform a manual check. This method does however still work if the camera zooms in (figure 30.) The matrix in figure 29 is the result.

$$\begin{bmatrix} 2 & 0 & -319 \\ 0 & 2 & -243 \\ 0 & 0 & 1 \end{bmatrix}$$

Figure 29: Matrix obtained from zooming in.

The above matrix signifies that the camera has zoomed in to approximately 2x zoom. The translation is a result of the frame having moved after the scaling.

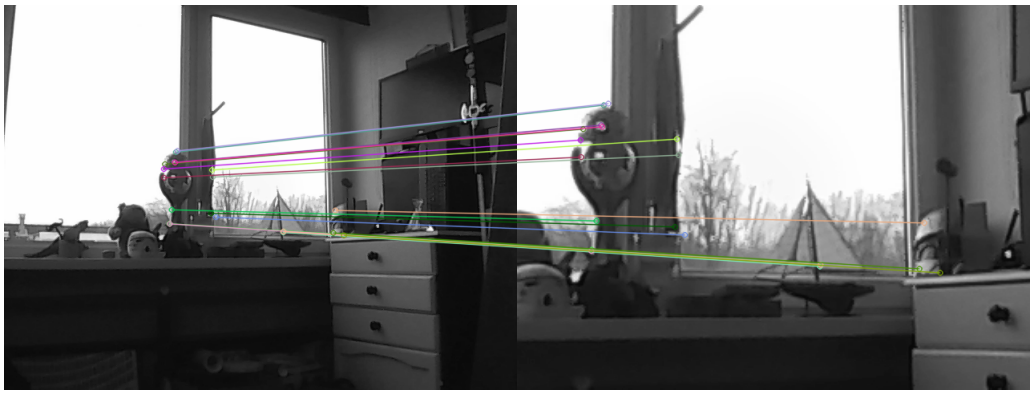

Figure 30: Matches found when image is zoomed in.

Because this is a digital zoom, an attacker can simply scale up part of the recorded image and send that to fool the system. If the camera supports optical zoom, it is possible to look for a change in perspective to verify if the image was actually zoomed in or only digitally enlarged.

5.2 ENF based detection

[1] Plotting the signal extracted from the audio of a forged recording against a recording of the ENF at the server gives the following figure 31. A high correlation can be seen between the camera stream and the recorded power, up until 300s at which point the attack starts.

[1] All results here are from [8]

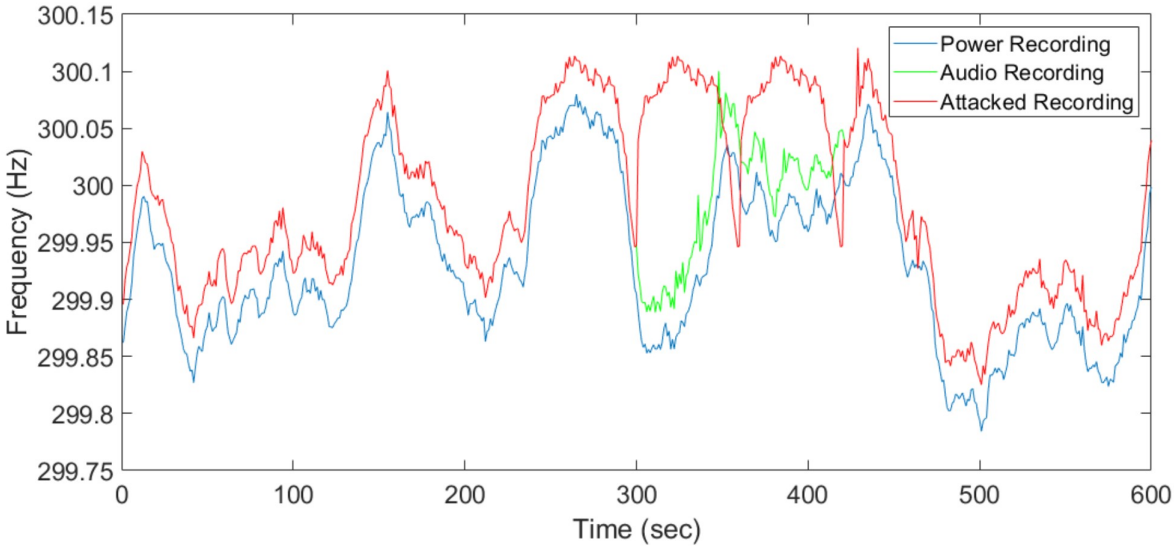

Figure 31: ENF Estimated from the power and original audio recording[8]

When correlating these two signals with a window size of 30s and a shift of 10s, it quickly becomes evident at which point the attacker started sending forged signals (figure 32). The method to determine the window size and shift are readable in [8].

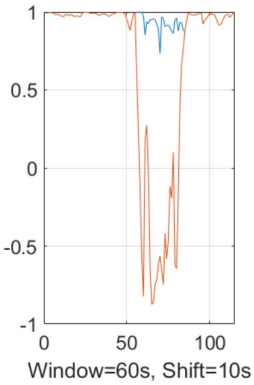

Figure 32: Result of Correlation[8]

5.3 Short discussion

While trying to replicate the implementation done by [8, 7], problems arose that prevented it from producing any results. Within the time-frame the source of the problem was not found. Possible sources are implementation, microphone, audio source, background noise, or a different issue. Due to time constrains it was decided to not try and find the problem but instead use the result found by [8] and explain their methods here as well in a shorter way. These results are also shown in the result section 16.

6 Conclusion

In this research project we looked at multiple detection methods and investigated their effectiveness against a frame duplication attack. We also took the opposite perspective, and investigated ways that an attacker might circumvent these detection methods. In conclusion using ENF based detection is the most effective detection method, and can be used on any camera system that has a microphone. The motion based detection can be used on any camera system that can move or zoom. While it is possible for an attacker to protect him/herself against these methods, most cameras do not possess the computational power to do this. Both methods are applicable to existing camera systems and do not require any kind of modifications to the cameras themselves. This only applies to cameras with either a microphone or an integrated servo.

6.1 Future Work

For future work it is useful to create a standard benchmark framework to test multiple detection methods for camera systems. Both in busy areas where the an frame duplication attack occurs if a certain person is in view as well as a empty hallway where the attack occurs if anything moves. This benchmark should also include changes in background (i.e. blinds close/open), and gradual and abrupt light changes.
A promising method is automatic timeline construction, where the server creates a timeline for each individual or object in view of the camera. This way if somebody simultaneously walks into a hallway and out the door an alert can be send to a guard.
For the ENF detection method specifically, more experiments with audio sources are needed to verify its effectiveness in busy areas where multiple electrical devices could be present, or where a lot of background noise is present.
Lastly a detection method needs to be found for cameras that are not able to move and do not have a microphone present. Currently some research is being done that looks into extrapolating the ENF through the intensity of TL-lighting, but with more light moving towards LED lighting this method becomes less interesting in commercial applications.

References

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features". In: *European conference on computer vision*. Springer. 2006, pp. 404–417.

[2] *ESP32-CAM Video Streaming and Face Recognition with Arduino IDE*. `https://randomnerdtutorials.com/esp32-cam-video-streaming-face-recognition-arduino-ide/`. Accessed: 4-2-2021.

[3] Johannes Kruse, Benjamin Schäfer, and Dirk Witthaut. "Predictability of Power Grid Frequency". In: *arXiv preprint arXiv:2004.09259* (2020).

[4] Brian Kulis and Kristen Grauman. "Kernelized locality-sensitive hashing for scalable image search". In: *2009 IEEE 12th international conference on computer vision*. IEEE. 2009, pp. 2130–2137.

[5] David G Lowe. "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2 (2004), pp. 91–110.

[6] Marius Muja and David Lowe. "Flann-fast library for approximate nearest neighbors user manual". In: *Computer Science Department, University of British Columbia, Vancouver, BC, Canada* 5 (2009).

[7] Deeraj Nagothu et al. "A study on smart online frame forging attacks against video surveillance system". In: *Sensors and Systems for Space Applications XII*. Vol. 11017. International Society for Optics and Photonics. 2019, p. 110170L.

[8] Deeraj Nagothu et al. "Detecting malicious false frame injection attacks on surveillance systems at the edge using electrical network frequency signals". In: *Sensors* 19.11 (2019), p. 2424.

[9] Henri J Nussbaumer. "The fast Fourier transform". In: *Fast Fourier Transform and Convolution Algorithms*. Springer, 1981, pp. 80–111.

[10] Kari Pulli et al. "Real-time computer vision with OpenCV". In: *Communications of the ACM* 55.6 (2012), pp. 61–69.

[11] *Rise of Surveillance Camera Installed Base Slows*. `https://www.sdmmag.com/articles/92407-rise-of-surveillance-camera-installed-base-slows`. Accessed: 4-2-2021.

[12] Ethan Rublee et al. "ORB: An efficient alternative to SIFT or SURF". In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2564–2571.

[13] *Transformation matrix*. `https://en.wikipedia.org/wiki/Transformation_matrix`. Accessed: 5-2-2021.

[14] Bilge Yesil. "Watching ourselves: Video surveillance, urban space and self-responsibilization". In: *Cultural Studies* 20.4-5 (2006), pp. 400–416.

A. Appendix

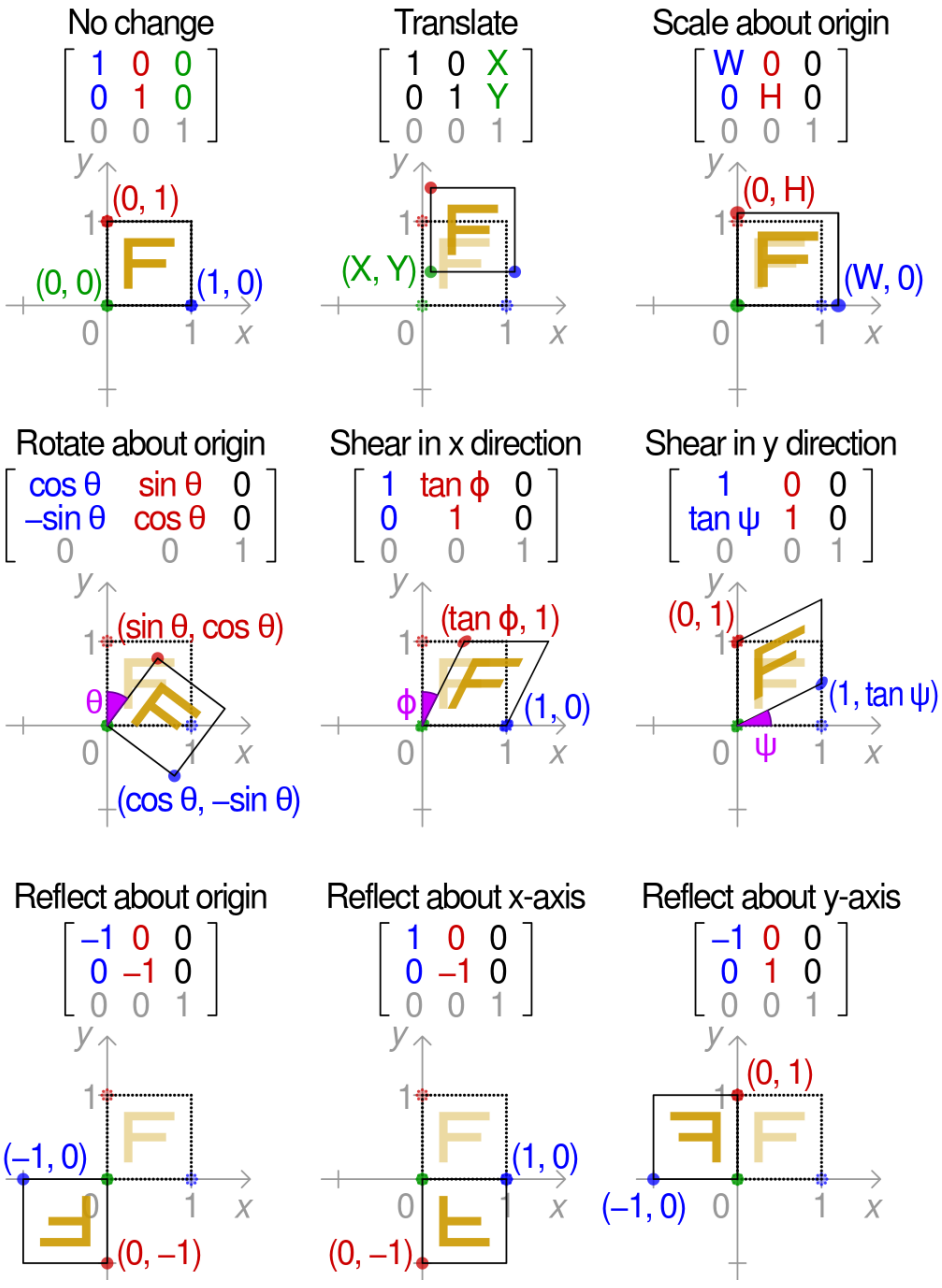

Figure 33: Complete operation set of an transformation matrix [13]