Live Video Forensics: Source Identification in Lossy Wireless Networks

Shaxun Chen[†], Amit Pande[†], Kai Zeng[‡], Prasant Mohapatra[†]
[†]Department of Computer Science, University of California, Davis, CA 95616
[‡]Department of Electrical and Computer Engineering, George Mason University, Fairfax, VA 22030
shaxunchen@gmail.com, pande@ucdavis.edu, kzeng2@gmu.edu, pmohapatra@ucdavis.edu

***Abstract*— Video source identification is very important in validating video evidence, tracking down video piracy crimes and regulating individual video sources. With the prevalence of wireless communication, wireless video cameras continue to replace their wired counterparts in security / surveillance systems and tactical networks. However, wirelessly streamed videos usually suffer from blocking and blurring due to inevitable packet loss in wireless transmissions. The existing source identification methods experience significant performance degradation or even fail to work when identifying videos with blocking and blurring. In this paper, we propose a method which is effective and efficient in identifying such wirelessly streamed videos. In addition, we also propose to incorporate wireless channel signatures and selective frame processing into source identification, which significantly improve the identification speed. We conduct extensive real-world experiments to validate our method. The results show that the source identification accuracy of the proposed scheme largely outperforms the existing methods in the presence of video blocking and blurring. Moreover, our method is able to identify the video source in a near-real-time fashion, which can be used to detect the wireless camera spoofing attack.**

***Keywords*— Video forensics, video source identification, wireless video streaming, video blocking**

I. INTRODUCTION

Source identification is a major component of video forensics. When presenting a video clip as evidence in a court of law, identifying the source (acquisition device) of the video is as important as the video itself. For instance, if a surveillance camera captured the scene of a suspect's alibi, it is necessary to prove that the video was truly recorded by the claimed camera. Otherwise it can be forged or derived from an untrustworthy source, which makes the evidence invalid. In the movie industry, significant revenue loss is caused every year by surreptitious recording in movie theaters and the subsequent illegal distribution. Video source identification is employed to track down such piracy crimes [1] [10]. While Internet enables video sharing at a large scale, it also opens doors for propagation of illegal or inappropriate materials, such as the video containing child porn or racial hatred. Video source identification can be used to regulate individual video sources [2].

Easier access to high-quality digital camcorders and sophisticated video editing tools urges the improvement of video source identification techniques. However, the research on this topic is still in its early stage. The most straight forward way to identify the video source is embedding digital watermarks into video when recording, but this method is computationally expensive and requires modification to recording devices, thus cannot be applied to most off-the-shelf cameras or camcorders. Some researchers proposed to utilize defective pixels (hot or dead pixels) on camera sensors to distinguish different devices. However, there are many cameras or camcorders do not have defective pixels. The most reliable method reported so far for source identification is based on the sensor pattern noise, which sources from the non-uniformity of each sensor pixel's sensitivity to light, and can be treated as the inherent fingerprint of a video capture device [3].

On the other hand, wireless communication has seen a tremendous growth in the recent years. Wireless cameras have also become increasingly popular. In the security camera market, wireless video cameras continue to replace their wired counterparts due to the ease of deployment. In tactical networks, wireless cameras are widely used as video sensors. They usually do not have local storage; video is captured and wirelessly streamed to a sink. Because of the inevitable packet loss and unpredictable transmission delay in wireless streaming, blocking and blurring frequently appear in the received frames. For such videos, experiments show that the existing source identification methods suffer from significant performance deterioration or even fail to work. Blocking and blurring caused by the lossy wireless channel severely tamper with the sensor fingerprint recognition.

Moreover, while the wireless camera brings great convenience, it also introduces a new type of security attack: wireless camera spoofing attack. An adversary can compromise a legitimate wireless camera, and then send fake video to the sink using the victim's identity. Since wireless cameras are mostly used in security systems and tactical networks, such an attack results in serious security threats. Wireless cameras are much more susceptible to this attack than their wired counterparts for a couple of reasons. First, tapping and sending through the air is much easier than invading physical cables. Second, wireless nodes are usually power constrained, and hereby cannot afford mass data (like video) encryption. Even if it is encrypted, the identity protection is still weak in wireless systems. For example, in 802.11 networks, even when WEP, WPA or WPA2 is enabled, management frames and control frames are still sent in plain text [4]. Fortunately, detecting the wireless camera spoofing attack can be viewed as a source identification task and thus borrow techniques from the existing methods. However, in this context, the identification should be performed in a much faster fashion, instead of the post-mortem solutions provided by the existing video forensics.

In this paper, we propose a systematic methodology for video source identification which can achieve excellent performance not only for the conventional videos but also for the wirelessly streamed videos with blocking and blurring. In addition, we propose to incorporate both the wireless signature and

sensor signature for video source identification. Together with other optimizations, we are able to identify the video source in a near-real-time fashion. This live video forensics not only performs the traditional source identification but also defends against the wireless camera spoofing attack. The contributions of this paper are as follows.

1) We improve the existing video source identification method, making it suitable for the videos contaminated by blocking and blurring. To the best of our knowledge, this is the first work trying to address the source identification problem in the context of wireless cameras.

2) We identify a security issue, wireless camera spoofing attack, which poses serious threat to security and surveillance systems. The existing body of research on wireless camera security is very small. The work presented in this paper is a contribution to the research of this area.

3) We enhance and expedite the video source identification method by incorporating wireless channel characteristics. Our method is able to identify the source device much faster than the existing methods, and therefore can be used to detect the camera spoofing attack in a timely manner.

Extensive real-world experiments are conducted to verify the effectiveness and efficiency of our method. The results show that our method largely outperforms the existing methods in the presence of video blocking and blurring. In addition, after incorporating the wireless characteristics, it is able to detect the camera spoofing attack within approximately 30 seconds.

The remainder of this paper is organized as follows. Section II discusses related work. Section III introduces our method for video source identification which tolerates video blocking and blurring, and Section IV improves its time efficiency and presents the procedure to detect wireless camera spoofing attacks. Section V evaluates our work and Section VI discusses the related issues. Section VII concludes the paper.

II. Related Work

The research on image source identification emerged a few years prior to video source identification, and often shares similar techniques with the latter. Kharrazi et al. [5] defined a set of image color features, such as average pixel values and RGB pairs correlation, and applied SVM to differentiate image capture devices based on these features. Similarly, Celiktutan et al. [7] defined a set of similarity measures with KNN and SVM used for classification. Choi et al. [6] further included lens radial distortions as part of the features. Popescu [8] developed the Expectation / Maximization algorithm to identify the demosaicing algorithm that a camera uses, based on which different image sources are classified. However, all these methods are only capable of telling the model or the manufacturer of the device, instead of identifying the individual camera that produced the image.

The following techniques focus on the specific device identification, which is desirable for forensic applications. Canon Data Verification Kit calculates the hash of images and uses a special secure memory card to enable tracing the image to a camera, but only high-end Canon DSLR cameras support this solution. Similarly, embedding watermarks into images is only applicable for specially designed devices rather than the off-the-shelf cameras. Geradts et al. [9] proposed to utilize sensor hot pixels or dead pixels to identify the image source. However, many cameras do not have such defective pixels, and newer products typically eliminate defective pixels by onboard post-processing. Lukas et al. [3] employed sensor pattern noise as an inherent fingerprint of the camera for source identification. Li [12] proposed to use adaptive weighting to improve the performance of [3]. The sensor pattern noise based schemes report the most reliable results so far, and will be further discussed in Section III.

Since a video consists of a sequence of frames, some methods mentioned above can be adopted directly for video source identification, such as [3], [5] and [9]. Su et al. [2] analyzed the motion estimation algorithms used by various video encoders to identify the source, but again, this method can only distinguish different models. Kurosawa et al. [11] measured the dark current noise of the sensor and used it as the device fingerprint. Since the dark current noise can only be extracted from dark frames, this method is restricted to the videos that contain dark frames. Chen et al. [1] inherited the idea proposed in [3] and applied it to videos. Houten et al. [13] and Hyun et al. [14] extended Chen's method for low resolution videos. However, all these solutions suffer from significant performance degradation when the video is captured by wireless cameras and contains blocking and blurring. Besides, most of them are computationally expensive and not suitable for fast identification.

Although wireless cameras are widely used for security purpose, very less attention has been paid to the security issues stemming from the wireless camera itself. Fogie [15] stated several potential vulnerabilities of wireless cameras, including the interception and substitution attack, but did not provide any solution. In this paper, we define the wireless camera spoofing attack, and also provide a systematic detection method to deal with such attacks.

A preliminary version of our work was reported in [23], which presented a video source identification technique tolerating blocking and blurring. In this paper, we further accelerate the identification process, and then apply the improved method to detecting wireless camera spoofing attacks.

III. Source Identification of Wirelessly Streamed Video

In this section, we first provide a brief background of video source identification, and then present the approach to extracting sensor pattern noise. After that, our source identification method is introduced, which is able to tolerate blocking and blurring, and efficiently identifies wirelessly streamed videos.

A. Background

Wireless video cameras (Figure 1a) are mostly used for security / surveillance purpose. The majority of commercially available products transmit via 802.11 channels, while the video sensors in tactical networks may use dedicated links. Some wireless cameras are AC powered, but others use rechargeable batteries or even solar power to go entirely untethered.

Wireless cameras do not have local storage; they capture the scene and stream to the sink in real time. The pixels of a video frame from a wireless camera can be presented as:

$$z_{ij} = L(y_{ij}) \qquad \text{where} \qquad y_{ij} = P(x_{ij}) \tag{1}$$

x_{ij} is the incoming light captured by the camera sensor at the pixel (i, j), where i = 1, 2, …, m and j = 1, 2, …, n. $m \times n$ is the sensor resolution. Here we assume the video resolution is equal to the sensor resolution for convenience. P indicates the sensor distortion and onboard processing. Sensor distortion consists of the non-uniformity of sensor pixels' sensitivity to light, shot noise, random noise and dark current; onboard processing includes (but is not limited to) intra-frame compression and inter-frame compression. Some cameras have more sophisticated onboard processing to improve the video quality.

(a) (b)

Figure 1. Wireless camera and frames with blocking and blurring

L is the distortion caused by wireless transmission. Because of the real-time requirement of security systems, typically UDP is used by wireless cameras for the video streaming and thus no retransmission occurs above the MAC layer. As a result, packet loss is the major factor of L. Packets with large jitter may have the same effect, because in a real-time system, if a packet fails to arrive in time, it does not contribute to video decoding. When a Cisco WVC80N wireless camera is set to 640×480 and MPEG4 best quality, as long as the frame rate is higher than 5 fps, significant amount of mosaic blocks appears in about 40 percent of the frames (see Figure 1b, work in 802.11n, normal office environment, 10 meters from the sink). This example shows, for wireless cameras, the blocking and blurring caused by packet loss (or large jitter) is not only inevitable but sometimes pervasive.

As mentioned above, video source identification methods fall into two categories: device class identification and specific device identification [16]. Only the latter, which is more difficult, meets the requirement of forensics use and the detection of camera spoofing attacks. Sensor pattern noise based method is so far the best choice in the latter category for the following reasons. First, this method is applicable for all videos. Some methods require specific devices (e.g. watermark based methods) while others have special limits on videos (e.g. [11]), but the sensor pattern noise based method exploits the inherent fingerprint of the camera sensor and is universally applicable. Second, this method has the best performance reported so far [1] [3]. In the next subsection, we will introduce how to extract the sensor pattern noise from a video.

B. Sensor Pattern Noise Extraction

Sensor pattern noise is mainly caused by the non-uniformity of each sensor pixel's sensitivity to light. Since for every frame of the video, various types of noises exist, such as white noise, shot noise and high-ISO noise, how to extract the sensor pattern noise is a challenge. Fortunately, most noises are likely to fade out after frame averaging. In different frames, the shot noise, ISO noise, white noise, etc. are randomly distributed, if we extract these noises from a large number of frames and then add together, they tend to cancel out. However, for the sensor pattern noise, it is the same for different frames (taken by the same camera), and going to be strengthened after being added up. Therefore, we can simply extract all the noise as a whole from each frame. The sensor pattern noise is supposed to survive the averaging while other noises would not [17].

Now we present the steps to extract the noise from a video frame [3] [18]. The basic idea is that for an image (a video frame), correlated signal is compressible and predictable but uncorrelated noise is not. We do not consider packet loss for now, which will be discussed in the next subsection.

1) Extract a frame from the video, denoted as F. Typically a frame has three color channels. Do steps *2) – 4)* for each channel. Given a channel, the data is viewed as a locally stationary i.i.d. signal plus a zero mean and stationary white Gaussian noise $N(0, \delta_0^2)$.

2) Calculate the fourth-level wavelet decomposition of the current channel with the 8-tap Daubechies wavelets. For every level, do steps *3)* and *4)*. For one fixed level, we denote the horizontal, vertical, and diagonal subbands as $h(i, j)$, $v(i, j)$, and $d(i, j)$, where (i, j) runs through an index set J that depends on the decomposition level.

3) For each subband (here, take d for example), estimate the local variance for each wavelet coefficient using MAP estimation. As to *local*, we use four sizes of a square $D \times D$ neighborhood E, where $D \in \{3, 5, 7, 9\}$.

$$\hat{\delta}_D^2(i, j) = \max[0, \frac{1}{D^2} \sum_{(i.j) \in E} (d^2(i, j) - \delta_0^2)] \qquad (i,j) \in J \tag{2}$$

$\hat{\delta}_D^2$ is the local variance estimation. We take the minimum of the four variances as the final estimate:

$$\hat{\delta}^2(i, j) = \min_D \hat{\delta}_D^2(i, j) \qquad D \in \{3, 5, 7, 9\}, (i,j) \in J \tag{3}$$

Perform the same operation for h and v subbands.

4) Obtain the denoised wavelet coefficients using Wiener filter:

$$d_{dN}(i, j) = d(i, j) \frac{\hat{\delta}^2(i, j)}{\hat{\delta}^2(i, j) + \delta_0^2} \qquad (i,j) \in J \tag{4}$$

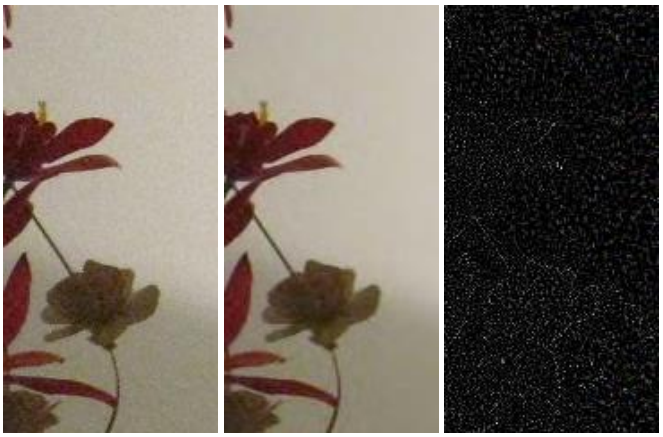
and similarly for h_{dN} and v_{dN}.

5) Based on the denoised wavelet coefficients obtained above (all four levels) and the fourth level low frequency subband, we can recover the denoised data in the current channel by inverse wavelet transform. Combine all three channels, we get the denoised frame F_{dN}. As a result, the extracted noise from frame F is:

$$N = F - F_{dN} \tag{5}$$

The above steps are implemented using Matlab. The frame is extracted from the video by *VideoReader*, and the *dwtmode* is set to *periodization*. Experiment shows δ_0 in Equation 2 and 4 (standard deviation of the white Gaussian noise assumed in the frame) does not have significant influence on the source identification accuracy. Here we choose δ_0 = 5. Figure 2 shows

an example of an original frame, the denoised frame (using the above method), and the noise extracted from this frame. For visualization, the noise (Figure 2c) is up scaled 10 times. This example is only for illustration purpose, in which the high-ISO noise dominates the sensor pattern noise, and some scene details are kept. We intentionally create this scene for illustration because in normal cases, the noise is too weak to bring any visible difference between F and F_{dN}.



(a) F (b) F_{dN} (c) $(F - F_{dN})\times 10$

Figure 2. Extracting noise from a frame

The noise extraction process will repeat for a sequence of frames from the same video. Based on our previous discussion, the sensor pattern noise will survive the averaging while other noises and minor scene details tend to cancel out. Therefore, the sensor pattern noise $\mathcal{N}$ can be expressed as:

$$\mathcal{N} = \frac{1}{k}\sum_{i}^{k} N_i \tag{6}$$

where N_i is the noise extracted from the i^{th} frame and k is the number of the frames processed. Typically, if the video quality is good, the sensor pattern noise can be well established when k is larger than 300 to 500. For source identification, we calculate the sensor pattern noise of the video to be identified (noted as $\mathcal{N}_v$), and compare it with the sensor pattern noise extracted from the camera (camera reference pattern for short, noted as $\mathcal{N}_c$) using the metric of correlation coefficient:

$$corr(\mathcal{N}_v, \mathcal{N}_c) = \frac{(\mathcal{N}_v - \overline{\mathcal{N}}_v)(\mathcal{N}_c - \overline{\mathcal{N}}_c)}{\left\|\mathcal{N}_v - \overline{\mathcal{N}}_v\right\|\left\|\mathcal{N}_c - \overline{\mathcal{N}}_c\right\|} \tag{7}$$

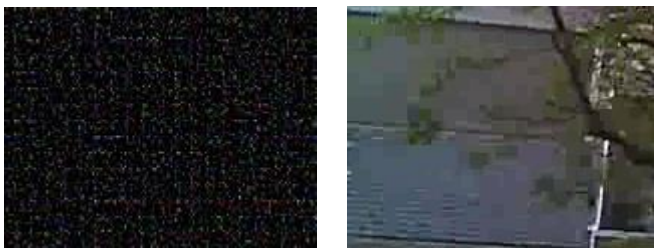
Calculating $\mathcal{N}_c$ is relatively easy because the source camera is typically accessible to law enforcers. We can use it to take a video (or multiple videos) with sufficient length and high quality, and thus derive the $\mathcal{N}_c$ accurately. However, as to the video to be identified, the length and quality are pre-given; we have to sit on what we have. Chen et al. reported that generally 40 seconds of video ($\geq$450kbps) is good enough for a reliable identification, but 10 minutes of video is required if the video quality is low (150kbps) [1]. Our experiments further show that, the videos contaminated by blocking and blurring are worse: even with decent bit rate (about 500kbps), in some cases, the $\mathcal{N}_v$ simply cannot converge; in other cases, it may require more than 20 minutes of video to get a decent accuracy using the existing method. It means that the wireless video evidence shorter than this length cannot be identified, which is not acceptable.

C. *Video Source Identification under Packet Loss*

In this subsection, we will improve the method presented above, making it suitable for detecting wirelessly streamed videos with blocking and blurring. We require successful identification by using a reasonable number of frames.

Video blocking affects the extraction of sensor pattern noise in two folds. First, within the blocks, the details as well as the pattern noise are lost. Typically when including more frames, the extracted sensor pattern noise will be strengthened. However, when adding a frame with blocking, within the blocking areas, the pattern noise would in fact be weakened (because in Equation 6, the denominator increases and numerator remains almost unchanged). Second, the borders of the blocking become a strong signal which will survive the extraction and averaging. As we know, videos are compressed based on 8×8 or 16×16 pixel blocks. Video blocking occurs if data of some blocks are missing. Therefore, the borders of the blocking areas only exist in the positions that are the integer multiples of 8 (or 16) pixels; eventually they form a "grid" instead of canceling each other, which will suppress the real sensor pattern noise. Figure 3a shows a sensor pattern noise contaminated by such "grid", where we can see many horizontal and vertical lines (they are not caused by *blockiness artifacts*, please refer to Section VI). It is extracted from a wirelessly streamed video of 360 frames with frequent packet loss, and it is up scaled 50 times for visualization purpose.

Video blurring can result from various reasons, such as high compression ratio, limited lens resolution or fast motion. But in this work, we only consider the blur caused by packet loss. As mentioned above, video is encoded based on blocks. Instead of losing all the data of a block, blurring is the result of losing high frequency component, but this information loss is still block based. That is, one block may look more blurred than another. By zooming in the red area in Figure 1b, we can see that the blur caused by packet loss is essentially smaller-size blocking (see Figure 3b). Therefore, we treat the blocking and blurring (caused by packet loss) uniformly thereinafter, which makes our method more clear and time efficient.



(a) "grid" interferes with pattern noise (b) zoom in the blur due to packet loss

Figure 3. Blockish effects in video

The basic idea of our method is as follows. For the frames with blocking, we rule out the blocking areas but still use the rest of the frame for pattern noise extraction. Such frames are not completely discarded because it would waste useful information and make the identification slower.

First, we introduce our method for blocking detection. The blocking detection methods can be divided into three categories: full reference, reduced reference, and no reference. Because we do not have the access to the original video without packet loss,

a non-reference method is required. The existing work, such as [19] and [20], are based on block boundary detection or Fourier transform. Since we have to perform the wavelet transform for noise extraction (see Section IIIB), we propose a blocking detection method that is also wavelet based. In this way, we can largely reduce the computational overhead, which is extremely important for the succeeding work in Section IV.

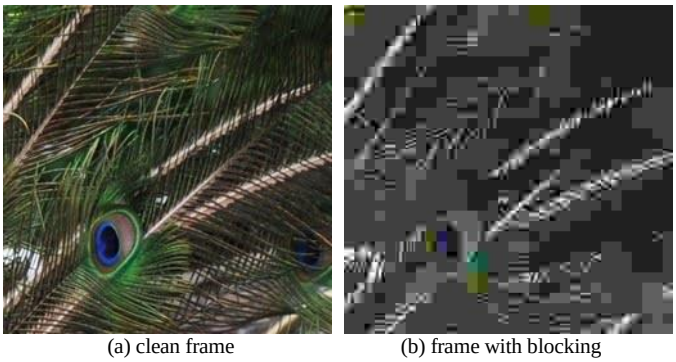
Taking a close look at the first-level wavelet transform result, we have two observations on the frames with heavy blocking:

1) More elements in the diagonal subband $d(i, j)$ are zeros or close to zero compared with clean frames.

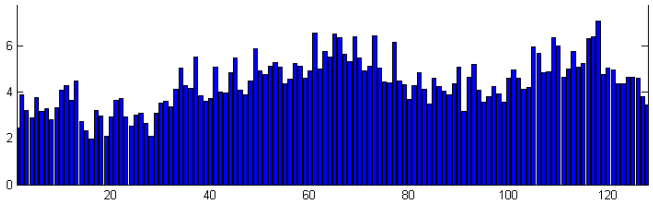
2) If we add the absolute values of $d(i, j)$ by rows (or by columns), the sum demonstrates periodic characteristics like the teeth of a saw. We will show the details below.

$$S_i = \frac{1}{p}\sum_{j}^{p} |\, d(i, j)\,| \qquad 1 \le i \le q \tag{8}$$

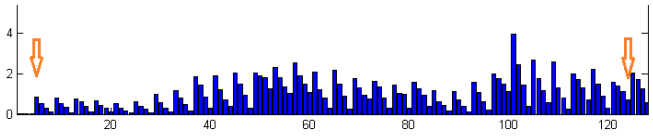
Equation 8 adds absolute values of the diagonal subband by rows and then takes an average. The case of adding by columns is similar. Here we assume the size of the frame (or part of the frame involved in the blocking detection) is $2p \times 2q$, and still use the 8-tap Daubechies wavelets as in Section IIIB.



(a) clean frame (b) frame with blocking



(c) S_i of the frame in (a), i = 1, 2, …, 128



(d) S_i of the frame in (b), i = 1, 2, …, 128

Figure 4. Obervations of $d(i, j)$ under video blocking

Figure 4 shows an example. Figure 4a is a clean frame and 4b is the same scene but with blocking. The S_i values of Figure 4a and 4b are shown in Figure 4c and 4d, respectively. From the plots, we can see S_i of Figure 4b have very small values and are significantly lower than those of the clean frame (i.e. observation 1). S_i in Figure 4d demonstrates a strong periodic pattern that, for every four bars (this period actually depends on the size of the smallest block; usually it is half of the codec square size), it goes from high to low (i.e. observation 2), which is not seen in Figure 4c. The plots of adding by columns (have not plotted due to limited space) have the same characteristics.

These two observations are not only true for this example, but generally applicable for the blocking areas in a video frame. The diagonal subband depicts the detail level of the original frame. Informally, if a pixel is substantially different from its adjacent pixels, the corresponding diagonal subband value tends to be high. Given a frame with blocking, inside a single block, obviously the detail level is low, while it is much higher at the border between different blocks. Therefore, when video blocking is heavy, this effect accumulates and will exhibit the visible pattern in S_i as described above. Here we want to point out that in our blocking detection, clean frames are not available as references, and we do not use any information from clean frames in our method. This example is only to show the difference of $d(i, j)$ between a clean frame and a frame with blocking.

Apparently, between these two observations, observation 1 is easier to exploit. But other than blocking, it is also true for big chunk of objects in plain color, such as a clean sky or white walls, which happen to be the best sources for extracting sensor pattern noise (they have fewer details, so sensor pattern noise is better preserved). We do not want to lose these high-quality sources when ruling out the blocking.

Fortunately, observation 2 is unique to video blocking and is our better choice. In order to detect the periodic pattern in S_i efficiently, we propose the following processing method. We maintain two cursors: one from the very left (i = 1), looking for the first local maximum, and another from the very right ($i = q$), seeking the first local minimum. Considering Figure 4d as an example, after this step, the left cursor should stop at bar 5 while the right one at bar 124 (marked by arrows in Figure 4d). Then, treating these two cursors as end points, we "fold" the x-axis in half (the bars out of the range between two end points are discarded). After folding, the values of overlapping bar pairs are added and then divided by 2, so we get:

$$S_j' = \frac{1}{2}(S_{v+j-1} + S_{w-j+1}) \qquad 1 \le j \le \frac{w-v+2}{2} \tag{9}$$

Here v is the index where the left cursor stops and w for the right one. j is a positive integer. We then compare the variance of S_j' and S_i:

$$\beta = \frac{\mathrm{var}(S_j')}{\mathrm{var}(S_i) + C_0} \qquad 1 \le j \le \frac{w-v+2}{2}, 1 \le i \le q \tag{10}$$

C_0 is a small positive constant to make sure the denominator is not zero. If β is lower than a threshold, we assume the periodic pattern (observation 2) is found, and thus report video blocking. The rationale behind is that if S_i is periodic, and within each period it is monotonic (complying with observation 2), S_j' calculated by Equation 9 should be much more evenly distributed than S_i. Here β reflects the degree of video blocking (the smaller, the heavier). The threshold is preset to 0.6, but for specific application scenarios, we can adjust the value and increase the detection accuracy. The performance and time costs of our blocking detection approach are given in Section VB.

Having introduced our approach for blocking detection,

now we briefly describe our improved version of the video source identification method.

A frame is divided into 32×32 pixel squares, where each square is parsed by the blocking detection algorithm presented above (it is equivalent to dividing $d(i, j)$ into 16×16 squares and calculating S_i separately). If the result is positive (video blocking exists), this square is discarded; otherwise, it is adopted for pattern noise extraction (using the method in Section IIIB). We should note that, when accumulating the noise from multiple frames, k in Equation 6 may have different values for each square. We use this method to calculate the sensor pattern noise for the video to be identified ($\mathcal{N}_v$). For the camera reference pattern ($\mathcal{N}_c$), usually blocking detection is not necessary because in this case video recording condition is under the control.

Next, Equation 7 is employed to judge whether $\mathcal{N}_v$ and $\mathcal{N}_c$ are from the same source or not. The correlation coefficient between $\mathcal{N}_v$ and $\mathcal{N}_c$ is typically from 0.1 to 0.7 when they have the same source. If the videos used to extract $\mathcal{N}_v$ and $\mathcal{N}_c$ are of the same bit rate, the correlation tends to be higher. Otherwise, it is slightly lower. When $\mathcal{N}_v$ and $\mathcal{N}_c$ are not from the same camera, $corr(\mathcal{N}_v, \mathcal{N}_c)$ is almost always lower than 0.01, where the gap is significant enough for us to perform identification (the threshold is noted as ψ). For videos with blocking and blurring, our method exhibits significant higher performance than the existing method without blocking detection. The experiment results will be shown in Section VC.

IV. Detecting Wireless Camera Spoofing Attack

In this section, we focus on expediting the method proposed in the last section. A near-real-time source identification method is provided for the detection of wireless camera spoofing attacks.

A. Attack Model

An attacker compromises a legitimate wireless video camera, and then sends fake video to the sink using the victim's identity, making the administrator believe it is still the video from the legitimate camera. We name such an attack wireless camera spoofing attack. Since wireless cameras are mostly used in security / surveillance systems, this attack can result in severe security threats.

As discussed in Section I, due to the "open air" nature of the wireless medium, camera spoofing attack is much easier to launch in wireless networks than wired networks. As we can see, this attack includes two essential steps: stealing the identity from a legitimate camera and disabling this camera, both of which are in fact not difficult to conduct. Taking 802.11 networks for example, even if WEP, WPA, or WPA2 are enabled, control frames and management frames are still unencrypted. The adversary can sniff the traffic and learn the MAC address of the legitimate user, and then masquerade as this user by modifying its own MAC address simply using an *ifconfig* command. In order to disable the legitimate user, the adversary can launch a deauthentication attack [4]. Two other methods to knock a wireless node off the air are given in [15].

We assume hardware-wise the attacker has the same ability as legitimate wireless cameras, such as the sensor resolution, video encoder and transmission frequency. Since in security / surveillance systems, cameras are usually mounted in specific spots, we assume they do not often change their locations.

B. Attack Detection

Since cryptography based authentication mechanism is not always feasible and secure [21], there is an increasing interest in using the physical layer information to detect such identity based attacks. Sensor pattern noise is a unique fingerprint that is able to differentiate an attacker from legitimate users; however, due to the high computational complexity, it is currently only used for post-mortem forensics analyses.

In the previous sections, we have mentioned some efforts to reduce the time cost of our method, such as dealing with the blocking and blurring uniformly, and reusing the calculated results of wavelet transform for blocking detection. Compared with the noise extraction, our blocking detection approach introduces very little extra overhead.

Nevertheless, processing a 640×480 video frame (including blocking detection and noise extraction) still takes about 10 seconds on an average laptop (with an Intel Core 2 Duo CPU). Being set to the highest video quality (1~1.5 Mbps), we need approximately 200 frames to make a reliable decision. The time it takes is too long for the spoofing attack detection. The attacker may have already achieved its goal when the attack is detected. Therefore, further improvements are required.

1) Parallelization. Simply shifting to a more powerful computer will not help much because the original approach is single-threaded. We parallelized the computationally expensive operations, such as wavelet transform and local variance estimation. With this modification, our method shows good scalability and the speed boosts accordingly in multi-core computers. For example, the modified version is more than 8 times faster when migrating from a 2-core laptop to a 2-CPU 12-core workstation (Xeon X5650×2). With an affordable multi-core computer, the typical identification time can be reduced from tens of minutes to a few minutes.

2) Selective frame processing. In a video clip, compared with P- and B-frames, an I-frame contains more fundamental information and details. By extracting noise exclusively from I-frames, we find that only 20~40 I-frames are able to achieve acceptable performance. More specifically, with about 40 I-frames, the performance is almost as good as 200 mixed frames; but with 20 I-frames, the result is marginal. We will explain soon how to improve the performance for the latter case.

3) Combining wireless fingerprints. We propose to incorporate wireless channel signatures with the sensor fingerprint for source identification. Profiles of wireless characteristics are built for each legitimate camera based on their history information. The metrics include: packet loss ratio, jitter, average signal strength, signal strength variance, and the percentage of blocking frames. The normalized values of these metrics are noted as $\vec{A} = (a_1, a_2, \ldots, a_5)$. The profile is time specific; we have 24 profiles for one camera, while each of them corresponds to one hour of a day.

$$\gamma(\vec{A}', \vec{A}_t) = \frac{(diag(\vec{\alpha}) \times \vec{A}') \cdot (diag(\vec{\alpha}) \times \vec{A}_t)}{\| diag(\vec{\alpha}) \times \vec{A}' \| \| diag(\vec{\alpha}) \times \vec{A}_t \|} \tag{11}$$

In Equation 11, $\vec{A}'$ is the measured wireless features of the current camera. $\vec{\alpha} = (a_1, a_2, \ldots, a_5)$ indicates the weights for each metric, which is predefined (see Section VD), and *diag* ($\vec{\alpha}$) means forming a diagonal matrix using $\vec{\alpha}$. Equation 11 basically calculates the cosine similarity between weighted $\vec{A}'$ and $\vec{A}_t$. Assuming $\vec{A}'$ is measured at time *t*, we choose the corresponding profile from 24 candidates, which is noted as $\vec{A}_t$. The larger γ we get, the more likely the camera is genuine. We also would like to mention that even if an attacker is placed at a location close to a genuine camera, as long as it is more than half of the wavelength away (e.g. 12.5cm for 2.4 GHz band), it will still have different channel characteristics from the genuine one due to the channel independency [22].

Now we describe our detection method for wireless camera spoofing attacks. We continuously perform the blocking detection and sensor pattern noise extraction (using the parallel version) on received I-frames. If the processing time of an I-frame is larger than the interval between two consecutive I-frames (CPU is too busy), we always select more recent ones to process and discard the older ones. Otherwise (CPU has idle time), we should set the video quality to a higher setting (the number of I-frames is supposed to increase and high quality videos are more difficult to spoof). If no higher setting is available, we opportunistically process extra P-frames.

The processing results of the most recent 20 I-frames (and possibly some P-frames) are used to calculate $\mathcal{N}_v$, and then compared with $\mathcal{N}_c$ by Equation 7. Instead of one threshold, now we use two-step thresholds. If $corr(\mathcal{N}_v, \mathcal{N}_c)$ is larger than ψ_1 (preset to 0.1), a genuine camera is assumed. If it is lower than ψ_2 ($\psi_2 < \psi_1$, preset to 0.01), an attack is reported. When the result is in between, we use γ calculated in Equation 11 to make the final decision. We always perform an incremental calculation, that is, use the noise extracted from the new I-frames to update $\mathcal{N}_v$, instead of calculating from scratch. With this method, we are able to perform an accurate camera spoofing attack detection within 30 seconds (using a 2-CPU workstation) even under heavy packet loss. More experiment results are shown in Section VD.

V. EVALUATIONS

A. Experiment Settings

TABLE I. CAPTURE DEVICES OF OUR EXPERIMENTS

Model	Amount	Sensor	Format	Net
Linksys WVC80N	4	640×480 CCD	MPEG4/ MJPEG	802.11n
D-Link 942L	1			
Axis M1011-W	1			
webcam of Lenovo X301	1			

The wireless cameras we use to evaluate our source identification method are listed in Table I. We choose these models because they are of the most popular brands in the market and have very similar specifications, which bring more challenges for source identification. Besides, all of them support both M PEG4 (with intra- and inter-frame compression) and MJPEG (intra-frame compression only). By default, we use MPEG4 videos unless otherwise specified. The sink is wiredly connected to a Cisco Linksys WRT160N V2 wireless-N router, to which the wireless cameras send the video. The webcam of X301 streams its video using VLC through the laptop's 802.11-n wireless network card.

B. Performance on Video Blocking Detection

The video blocking detection algorithm described in Section IIIC is the basis of our source identification method. We propose this new blocking detection technique mainly due to the considerations of time efficiency. In this subsection, we evaluate the accuracy and time cost of our blocking detection approach.

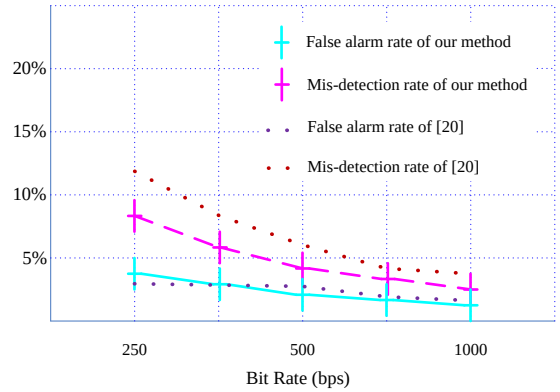


Figure 5. Accuracy of our blocking detection approach

The false alarm and mis-detection rate of our approach are shown in Figure 5. False alarm is defined as the video blocking reported by our approach while there is actually none; mis-detection refers to the cases where over 50 percent area of the square is occupied by blocking but our approach fails to detect. A total number of 6000 squares (32×32 pixel units in frames) are calculated (about 2300 of them are blockish), which are evenly collected from the cameras listed in Table I. The x-axis shows the codec bit rate (MPEG4, bit rates are approximate values; for the cameras whose bit rates cannot be set explicitly, we estimate them by the file size). The threshold of β is set to 0.6. The ground truth is counted manually. From the results we can see that both the false alarm and misdetection rates are very low, especially for high bit rate videos, which is sufficient for our succeeding processing. We also compare the accuracy of the method provided in [20] with ours. The performance trends are similar and our method has slightly higher accuracy.

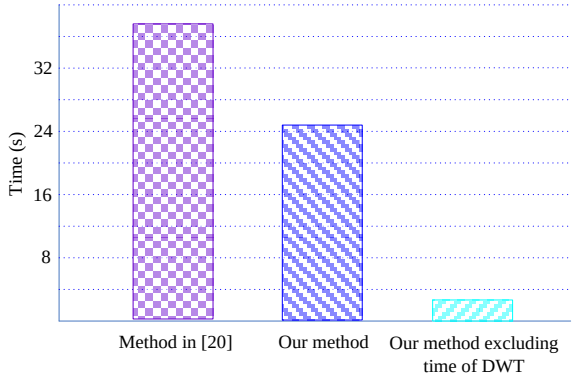


Figure 6. Time efficiency of our blocking detection approach

Figure 6 shows the time cost of our blocking detection approach for ten 640×480 frames using an average laptop. The method provided in [20] is referred as the baseline. The third bar plots the time of our approach without including the time spent on wavelet transform (since it has been done during the noise extraction, we do not need to calculate the wavelet transform again). However, if using the existing blocking detection methods, which are not wavelet based, we cannot save any time (the first bar). By developing our own blocking detection approach, we speed up about 15 times.

C. *Performance on Video Source Identification*

We first testify the effectiveness of our source identification method. The basic requirement is that, with sufficient length of videos, the sensor pattern noise extracted from different source devices should be clearly distinguishable, and the ones shared the same source should exhibit high correlation.

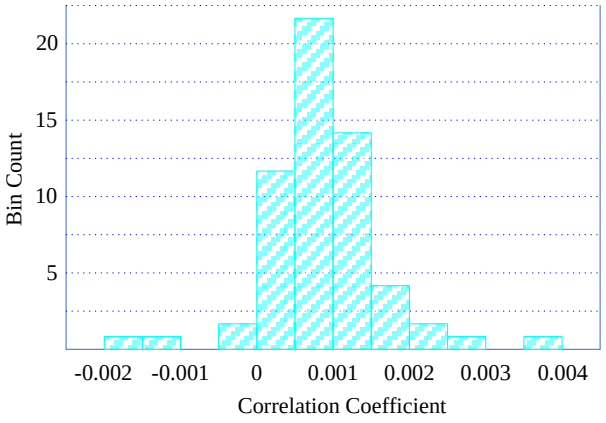


Figure 7. Correlation between blockish videos from different sources

We use a Linksys WVC80N camera as the reference, and take a 20-minutes-long video at the highest quality in order to extract an accurate $\mathcal{N}_c$. Then, we shoot 60 videos using other 6 cameras (10 videos each), all of which are in high quality and 3 minutes long (with heavy blocking). More specifically, all the videos are in MPEG4 format, and the average data rates of the videos collected from 4 different models are 1.55 Mbps, 1.63 Mbps, 1.75 Mbps, and 1.60 Mbps, respectively (in the same order as in Table I). The data rates are calculated by file size and video length. We try our best to adjust the settings of these video cameras (most of them do not allow specifying bit rate directly), to make all the models working at approximately the same data rate. Figure 7 plots the correlation coefficient between $\mathcal{N}_v$ of these 60 videos (calculated by our method) and $\mathcal{N}_c$ of the reference camera, from which we can see the correlations are very low (the absolute values are mostly less than 0.003).

Next, we use the same camera as the reference, but take another 30 videos of various scenes by this camera (also in high quality, with heavy blocking, and 3 minutes long). Since $\mathcal{N}_v$ and $\mathcal{N}_c$ are now from the same camera, the correlations between them are high (see bars in Figure 8), which is an obvious contrast against Figure 7. The solid curve in Figure 8 is an approximation of the bars; the same approximations for lower-bit-rate videos are plotted in the dashed curves (the data rates shown in the legend of Figure 8 are approximate values; but similarly as in the last experiment, we carefully adjust the settings to make all the cameras work at approximately the same bitrates). Although the correlation slightly decreases for lower rates, it still demonstrates huge difference compared with Figure 7. The results show that the sensor pattern noise is an effective signature of video sources.

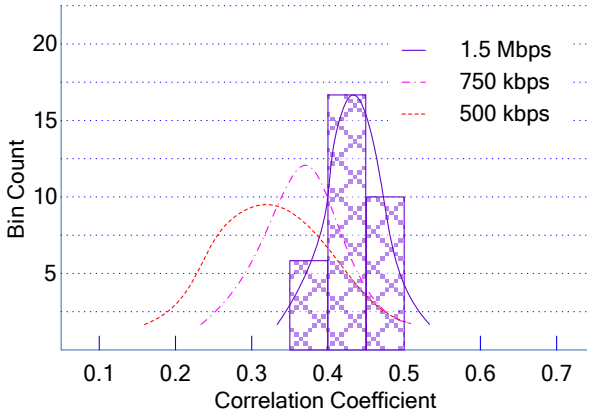


Figure 8. Correlation between blockish videos from the same source

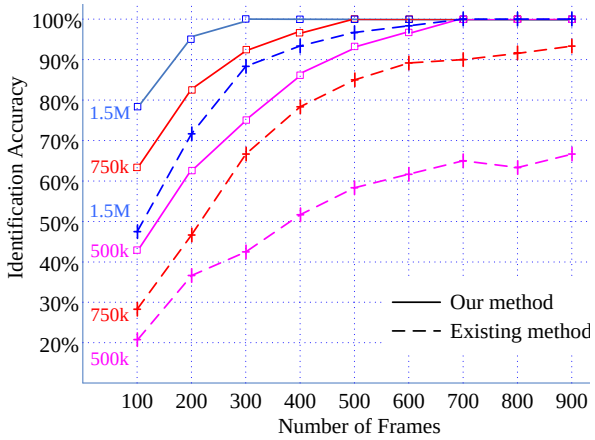


Figure 9. Identification accuracy under light blocking

Now we compare the performance of our method with the existing source identification method. For our method, we only use one threshold and let $\psi = 0.01$, that is, if $corr(\mathcal{N}_v, \mathcal{N}_c)$ is larger than 0.01, we assume the video to be identified are from the same source as $\mathcal{N}_c$; otherwise it is not. Considering the great discrimination shown in Figure 7 and 8, this simple strategy is good enough to make the decision. Figure 9 presents the comparison results when only light blocking occurs (less than 20% of the frames are contaminated by blocking and blurring). Three bit rates are tested respectively. For each rate, 140 videos are collected (20 from each camera). All videos are 640×480 and 10 fps. Accuracy is defined as the correct identifications divided by the total number. x-axis denotes the number of frames used for $\mathcal{N}_v$ extraction. The solid lines show the performance of our method while the dashed lines are from the existing method (provided in [1], without blocking detection). We can see that under the same bit rate, our method achieves the perfect accuracy with much less frames. For low bit rates, the

gap is even larger: videos (with blocking) cannot be accurately identified by the existing method even if sufficient frames are provided; instead, our method gives satisfactory results.

In Figure 10, we present the performance comparison under the heavy blocking (about 40% of the frames are contaminated by blocking). For 6400 and 12800 frame cases, we test 70 videos; other settings are the same as the test above. The result clearly shows, under heavy blocking, the existing method cannot achieve acceptable performance even if the video is long enough (because the borders of the blocking area suppress the sensor pattern noise, see Section IIIC). Our method, in contrast, exhibits near-linear accuracy improvement as video length increases.

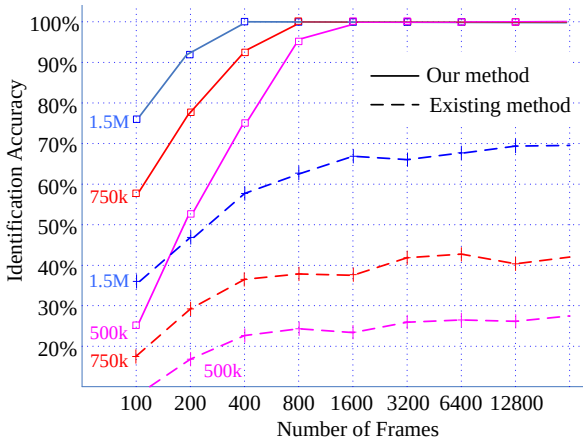


Figure 10. Identification accuracy under heavy blocking

The next experiment presents how the performance of our method is influenced by the heaviness of video blocking. The performance of the existing method under the same condition is also tested for the comparison purpose. In Figure 11, x-axis shows the percentage of the frames contaminated by blocking and blurring, while y-axis is the identification accuracy (using 400 consecutive frames). The bit rates of the videos in this experiment are all 750 kbps. Other settings are the same as above. Since controlling the percentage of the blocking frames is very difficult, we first vary the locations of wireless cameras, collecting a large amount of videos, and then divided them into 5 groups according to the heaviness of their blocking. The percentages shown in x-axis are approximate values (but are the same for two different methods).

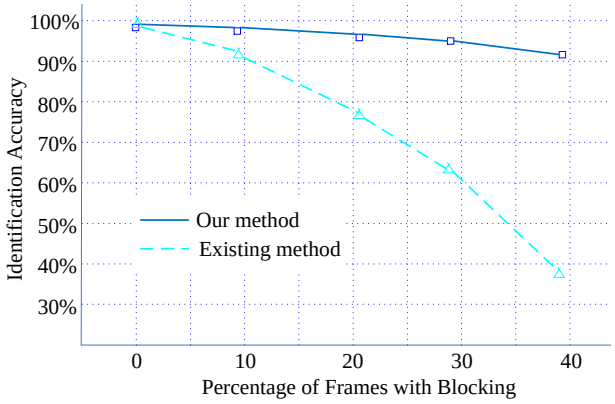


Figure 11. Performance influenced by the percentage of blocking frames

From the result we can see that with no or very little blocking, our method has almost the same performance as the traditional method. However, when the blocking frames are more than 10 percent of the total frames, the performance of the traditional method decreases dramatically, while our method only experiences a slight performance drop. More important, with the help of more frames (this experiment only use 400 frames), our method can always achieve perfect accuracy even the blocking is very heavy (Figure 10).

D. *Performance on Camera Spoofing Attack Detection*

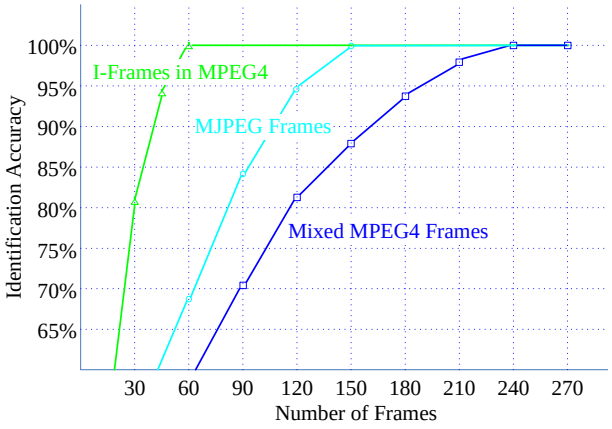


Figure 12. Identification using different frames

In this subsection, we evaluate the performance of our detection method for wireless camera spoofing attacks. In the following experiments, we use three identical laptops (with built-in webcam) acting as the legitimate node as well as attackers due to the easier access to physical layer information (i.e. wireless signal strength). The open source software VLC is employed to stream the video.

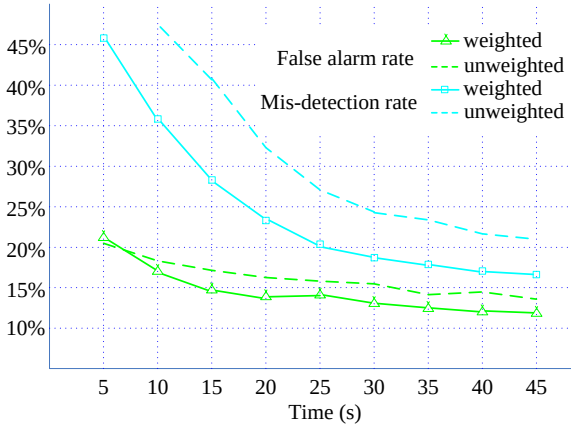


Figure 13. Identification with merely wireless characteristics

We first test the contribution of different frames to the sensor pattern noise extraction. MPEG4 frames (all frames), MJPEG frames, and pure I-frames from MPEG4 are compared in Figure 12 (all set to high quality, light blocking). x-axis shows the number of frames used. The results show that for MJPEG, fewer frames are required to achieve the same performance as MPEG4, which is easy to understand because MJPEG does not have inter-frame compression. Interestingly, pure I-frames are

even more efficient than MJPEG frames. This is probably because, compared to the latter, the scene change between two consecutive I-frames is more dramatic (there are a number of P- or B-frames in between), so that the scene detail residue is easier to cancel out during the noise averaging.

As described earlier, we incorporate five types of wireless channel characteristics for source identification. Here, we evaluate the identification accuracy they can provide without using sensor pattern noise. Both the legitimate user and attackers have the same software and encoding settings (MPEG4, 1.6Mbps). One attacker is placed close to the legitimate node and the other is far away. In each test, we perform 20 trials, and then change their locations (both the good node and attackers) for the next test. 10 tests are performed (200 trials) and the averages are plotted in Figure 13. They are all conducted in a crowded office environment (with intensive WiFi signals). x-axis denotes the time duration that wireless characteristics have been collected. Dashed lines show the unweighted case, i.e., $\vec{\alpha} = I_5$ (see Section IVB). We test various weight combinations, and find $\vec{\alpha}$ = (1, 1, 2, 2, 1) has relatively the best performance, which is plotted as solid lines in Figure 13.

Now we combine the wireless characteristics and the sensor pattern noise for wireless camera spoofing attack detection. We use the two-step thresholds discussed in Section IVB and $\vec{\alpha}$ is set to (1, 1, 2, 2, 1). 400 trials are performed (all in MPEG4, high quality) at 10 different locations (7 of them suffer from heavy packet loss). Other settings are the same as the last experiment. The results are plotted in Figure 14. We can see that our method can achieve very high detection accuracy (very low false positive rate and false negative rate) within about 30 seconds (in a workstation with Xeon X5650×2), during which about 20 I-frames are processed. As a comparison, the existing method needs tens of minutes to collect video frames and extra time for frame processing, while still has a much lower accuracy (see Figure 10). The results indicate that our method can be used to defend against wireless camera spoofing attacks in a near-real-time fashion.

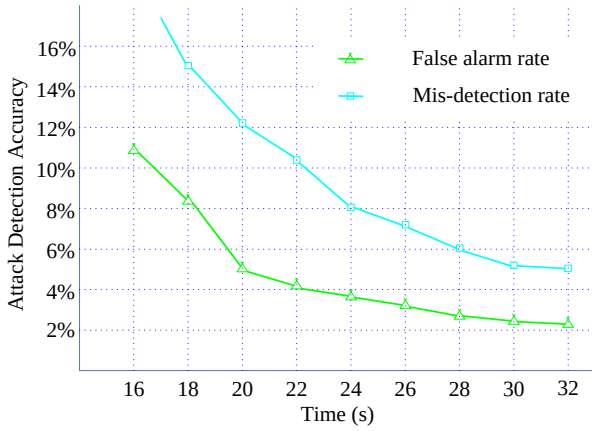


Figure 14. Performance of spoofing attack detection

VI. Discussion

In this paper, we treat two terms *video blocking* and *blockiness artifacts* differently. The former refers to the mosaic blurring in videos due to packet loss (see Figure 1b), while the latter is the periodic signal caused by the block based compression scheme of video codec, which is always there even without any blur or packet loss. In our work, we use the similar approach as in [1] (but ours is wavelet based) to remove blockiness artifacts before blocking detection and noise extraction. So the lines in Figure 3a are mainly caused by video blocking instead of blockiness artifacts. We did not mention this step earlier because it may introduce unnecessary confusions.

All the videos in our evaluations are in the size of 640×480. With minor modifications, our method can also apply to the identification of the videos that are smaller than the sensor's native resolution. The main focus of this paper is to provide a solution to the wireless streaming and video blocking related issues in source identification. For different formats, resolutions, gamma correction, etc., there are a number of existing studies out there [1] [3], which are orthogonal to our research and can be incorporated into out method.

VII. Conclusion

In this paper, we introduced a systematic method for source identification of wirelessly streamed videos. Our method exhibits excellent performance even in the presence of video blocking and blurring. To achieve this goal, we developed a novel blocking detection approach. We also proposed to incorporate wireless channel signatures and selective frame processing to accelerate our method, which enables the near-real-time video source identification. Real-world experiments are conducted to evaluate the effectiveness and efficiency of our method. The results show that the proposed approach largely outperforms the existing method in both accuracy and time efficiency.

References

[1] M. Chen, J. Fridrich, M. Goljan, J. Lukas, "Source digital camcorder identification using sensor photo response non-uniformity," SPIE International Conference on Security, Steganography, Watermarking of Multimedia Contents, vol. 6505, no. 1, 2007.

[2] Y. Su, J. Xu, B. Dong, "A source video identification algorithm based on motion vectors," International Workshop on Computer Science and Engineering, vol. 2, pp. 312-316, 2009.

[3] J. Lukas, J. Fridrich, M. Goljan, "Digital camera identification from sensor pattern noise," IEEE Transactions on Information Forensics and Security, 1(2): 205-214, 2006.

[4] J. Bellardo, S. Savage, "802.11 Denial-of-service attacks: real vulnerabilities and practical solutions," 12th Conference on USENIX Security Symposium, pp. 15-28, 2003.

[5] M. Kharrazi, H. Sencar, N. Memon, "Blind source camera identifyication," IEEE International conference on Image Processing, Singapore, 2004.

[6] K.S. Choi, E. Lam, K. Wong, "Automatic source camera identification using the intrinsic lens radial distortion," Optics Express, 14(24): 11551-11565, 2006.

[7] O. Celiktutan, I. Avcibas, B. Sankur, N. Memon, "Source cell-phone identification," International Conference on Advanced Computing and Communication, Tamil Nadu, India, 2005.

[8] A.C. Popescu, "Statistical tools for digital image forensics," Ph.D dissertation, Department of Computer Science, Dartmouth College, 2004.

[9] Z. Geradts, J. Bijhold, M. Kieft, K. Kurosawa, N. Saitoh, "Methods for identification of images acquired with digital cameras," Enabling Technologies for Law Enforcement and Security, pp. 505-512, Feb 2001.

[10] F. Lefebvre, B. Chupeau, A. Massoudi, E. Diehl, "Image and video fingerprinting: forensic applications," SPIE-IS&T Electronic Imaging, vol. 7254, article 725405, pp. 1-9, 2009.

[11] K. Kurosawa, K. Kuroki, N. Saitoh, "CCD fingerprint method - identifi-

cation of a video camera from videotaped images," IEEE International Conference on Image Processing, pp. 537-540, 1999.

[12] C.T. Li, "Source camera identification using enhanced sensor pattern noise," IEEE Transactions on Information Forensics and Security, 5(2): 280-287, 2010.

[13] W. van Houten, Z. Geradts, "Source video camera identification for multiply compressed videos originating for YouTube," Digital Investigation, 6(1-2): 48-60, 2009.

[14] D.K. Hyun, C.H. Choi, H.K. Lee, "Camcorder identification for heavily compressed low resolution videos," Lecture Notes in Electrical Engineering, vol. 114, part 5, pp. 695-701, 2012.

[15] S. Fogie, "Abusing and misusing wireless cameras," http://www.informit.com/articles/article.aspx?p=1016099, 2007.

[16] A. Rocha, W. Scheirer, T. Boult, S. Goldenstein, "Vision of unseen: Current trends and challenges in digital image and video forensics," ACM Computing Surveys, 43(4) article 26, pp. 1-42, 2011.

[17] G.C. Holst, CCD Arrays, Cameras and Displays, JCD Publishing, 1998.

[18] M.K. Mihcak, I. Kozintsev, K. Ramchandran, "Spatially adaptive statistical modeling of wavelet image coefficients and its application to denoising," IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 6, pp. 3253-3256, 1999.

[19] J. Yang, H. Choi, T. Kim, "Noise estimation for blocking artifacts reduction in DCT coded images," IEEE Transactions on Circuits and Systems for Video Technology, 10(7): 1116-1120, 2000.

[20] Z. Wang, A.C. Bovik, B.L. Evan, "Blind measurement of blocking artifacts in images," IEEE International Conference on Image Processing, Vancouver, Canada, 2000.

[21] K. Zeng, K. Govidan, D. Wu, P. Mohapatra, "Identity based attack detection in mobile wireless networks," IEEE Infocom, Shanghai, China, 2011.

[22] R. Wilson, D. Tse, R.A. Scholtz, "Channel identification: secret sharing using reciprocity in ultrawideband channels," IEEE Transactions on Information Forensics and Security, 2(3): 364-375, 2007.

[23] S. Chen, P. Amit, K. Zeng, P. Mohapatra, "Video source identification in lossy wireless networks," IEEE Infocom, 215- 219. Turin, Italy, 2013.

Shaxun Chen is currently a research scientist at Facebook. He received his Ph.D. degree from the Department of Computer Science at University of California, Davis, in 2014. Before that, he obtained his B.Sc and M.Sc degrees in Computer Science from Nanjing University, China, in 2005 and 2008, respectively. He has published more than 10 papers in top-tier journals and conferences, including MobiSys, IEEE/ACM Transactions on Networking, ICNP, Infocom, and IEEE Transactions on Mobile Computing. Dr. Chen's research interests span a wide range of topics in network security, wireless traffic analysis, human-computer interaction, and smartphone user profiling.

Amit Pande is currently a research scientist in the Department of Computer Science, University of California, Davis. He completed his PhD from Iowa State University in 2010 on designing algorithms and architecture for secure embedded multimedia systems. He was a receipient of the NSF Computing Innovation Fellowship (2010-2012). He won the Excellence in Postdoctoral Research Award at UC Davis in 2012. Dr. Pande's research interests include mobile security, applied machine learning, and video applications.

Kai Zeng is currently an assistant professor in the Department of Electrical and Computer Engineering and affiliated with the Center for Secure Information Systems at George Mason University. He received his Ph.D. degree in Electrical and Computer Engineering at Worcester Polytechnic Institute (WPI) in 2008. He was a postdoctoral scholar in the Department of Computer Science at University of California, Davis (UCD) from 2008 to 2011, and an assistant professor in the Department of Computer and Information Science at University of Michigan - Dearborn from 2011 to 2014. Dr. Zeng was a recipient of the U.S. National Science Foundation Faculty Early Career Development (CAREER) award in 2012. He won Excellence in Postdoctoral Research Award at UCD in 2011 and Sigma Xi Outstanding Ph.D. Dissertation Award at WPI in 2008. He was a visiting faculty of Air Force Research Laboratory through Visiting Faculty Research Program (VFRP) in summer 2013. Dr. Zeng's current research interests are in cyber physical system security and privacy, network forensics, physical layer security, and cognitive radio networks.

Prasant Mohapatra is a professor in the Department of Computer Science and is serving as the Associate Chancellor of the University of California, Davis. He was the Department Chair of Computer Science during 2007-2013, and held the Tim Bucher Family Endowed Chair Professorship during that period. He served as the Interim Vice-Provost and the Campus CIO of UC Davis during 2013-2014. In the past, he has been on the faculty at Iowa State University and Michigan State University. He has also held Visiting Scientist positions at Intel Corporation, Panasonic Technologies, Institute of Infocomm Research (I2R), Singapore, and National ICT Australia (NICTA). He has been a Visiting Professor at the University of Padova, Italy and Yonsei University, and KAIST, South Korea.

Dr. Mohapatra is the Editor-in-Chief of the IEEE Transactions on Mobile Computing. He has served on the editorial board of the IEEE Transactions on Computers, IEEE Transactions on Mobile Computing, IEEE Transaction on Parallel and Distributed Systems, ACM WINET, and Ad Hoc Networks. He has served as the Program Chair and the General Chair and has been on the program/organizational committees of several international conferences. He has been a Guest Editor for IEEE Network, IEEE Transactions on Mobile Computing, IEEE Communications, IEEE Wireless Communications, and the IEEE Computer.

Dr. Mohapatra received his doctoral degree from Penn State University in 1993, and received an Outstanding Engineering Alumni Award in 2008. He also received an Outstanding Research Faculty Award from the College of Engineering at the University of California, Davis. He received the HP Labs Innovation awards in 2011, 2012, and 2013. He is a Fellow of the IEEE and a Fellow of AAAS.

Dr. Mohapatra's research interests are in the areas of wireless networks, mobile communications, cybersecurity, and Internet protocols. His research has been funded through grants from the National Science Foundation, US Department of Defense, Army Research Labs, Intel Corporation, Siemens, Panasonic Technologies, Hewlett Packard, Raytheon, Huawei Technologies, and EMC Corporation.