Received February 7, 2020, accepted March 4, 2020, date of publication March 23, 2020, date of current version April 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2982620

Remote Check Truncation Systems: Vulnerability Analysis and Countermeasures

HAFIZ MALIK[1], (Senior Member, IEEE), RIGEL GJOMEMO[2], V. N. VENKATAKRISHNAN[2], RASHID ANSARI[3], (Life Fellow, IEEE), AND AUN IRTAZA[1]

[1]Department of Electrical and Computer Engineering, University of Michigan–Dearborn, Dearborn, MI 48128, USA
[2]Department of Computer Science, University of Illinois at Chicago, Chicago, IL 60607, USA
[3]Department of Electrical and Computer Engineering, University of Illinois at Chicago, Chicago, IL 60607, USA

Corresponding author: Hafiz Malik (hafiz@umich.edu)

ABSTRACT All major banks in the USA and around the world offer remote check deposit services. Consumers can use their smart phones to deposit checks remotely. This new online check truncation system is vulnerable to a wide range of attacks, including *digital check forgery*. Shifting trust from a human teller or an automated teller machine (ATM) to a smart device (cell phone) provides new attack surfaces. This paper exploits security vulnerabilities in the existing remote check deposit system and presents an attack vector for existing remote check truncation systems. The proposed attack vector exploits vulnerabilities in the untrusted client-side check-deposit system that enables an attacker to instrument the check deposit application library. The instrumented library allows the attacker to induce digital check forgery with minimized tampering artifacts. It has been observed through this investigation that digital check forgery-based attacks are more powerful than conventional paper-based check forgery attacks. The effectiveness of these attacks is evaluated by targeting three leading banks in United States, finding that all three of the targeted banks are vulnerable to the proposed attacks. A set of countermeasures based on digital check verification is also proposed to combat digital check forgery attacks on existing remote check deposit systems. The proposed countermeasures rely on tamper detection in digital images and expert-system based decision fusion. The effectiveness of the proposed framework is evaluated using tampered check images. The tampered images used for performance evaluation also include set of tampered images used for successfully attacking the remote check deposit systems(being using by leading banks around the world today). Experimental results show that the proposed expert system-based framework is capable of detecting digital check forgery attacks.

INDEX TERMS Check truncation system, online banking, remote check deposit, digital check forgery, forgery detection, image forensics, expert system, library instrumentation, JPEG artifacts.

I. INTRODUCTION

Today, all major banks in the USA and around the world offer remote check deposit services using smartphones and scanning-equipped computing devices connected to the internet. The number of customers using these remote check deposit services is on the rise. For instance, according to a study on the state of remote deposit [1], [2], by the end of 2016 around 40% of small businesses were using mobile remote check deposit (mRCD) and around 20% of customers were also using (mRCD) services. The remote check deposit process is an alternative to the commonly used paper-based check truncation system. The paper-based check truncation system requires the physical presence of the customers to go to either an automated teller machine (ATM) or a banking station to deposit the check to a bank teller. The US Check 21 Act [3] is the main driving force behind the development and deployment of the remote check deposit process. The equivalence between a paper check and its electronic representation (i.e., digital images) was established by the US Check 21 Act [3]. The US Check 21 Act [3] also regulated the *digital check truncation* process, which replaces a paper check with its electronic representations (e.g., digital image of the check) for the check deposit and clearing process. Financial efficiency (e.g., cost reduction) related to paper-based check processing among financial institutions is one of the motives behind the US Check 21 Act [3].

Shown in Figure 1 is an outline of the remote check deposit process. It consists of check scanning, digital image analysis, and check deposit and clearing subsystems. The *client*

The associate editor coordinating the review of this manuscript and approving it for publication was Jiafeng Xie.

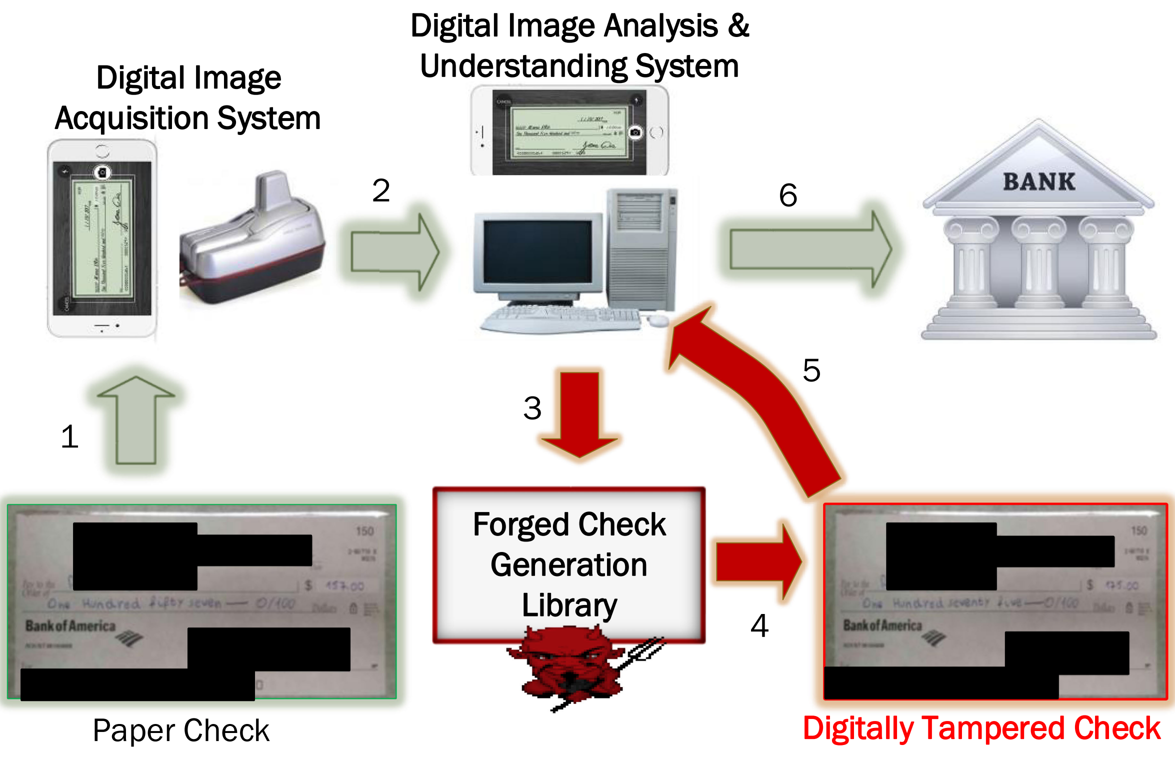

Digital Image Acquisition System
Digital Image Analysis & Understanding System
BANK
1
2
3
4
5
6
Forged Check Generation Library
Paper Check
Digitally Tampered Check

FIGURE 1. **Illustration of remote check truncation system and proposed attack model.**

truncation system (e.g., remote check deposit App) shown in Figure 1 is capable of:

1) Digital Image Acquisition—a device capable of scanning and/or acquiring digital images of both sides of the paper check. It consists of check scanner or a smartphone.
2) Digital Image Analysis—a subsystem capable of analyzing and processing digital check images. It consists of a computer program and/or smartphone App
3) Communication Subsystem—a subsystem on the client device to transmit acquired images, the output of the image analysis subsystem, and information associated with the check deposit and clearing process over the internet to the servers for check clearing

Subsequently, the server side processes and analyzes the received digital check images using an optical character recognition (OCR) operation to extract the amount and the routing and account numbers information. The extracted check information, e.g., amount, bank routing number, and bank account number are then used for check clearing. Two different realizations of remote check deposit are being used today: one is for business use and the second one is for personal use. A remote check deposit system for business use consists of a scanning and computing system, whereas, a personal use version consists of a remote check deposit App installed on a smartphone. The use of remote check deposit is on the rise due to its salient features such 24/7 availability, cost reduction, convenience, digital record keeping, etc. Recent studies indicate that millions of customers in the United States and around the world are using mRCD everyday [4]–[8].

Recent studies have highlighted that paper-based check deposits are the most vulnerable to financial frauds among all payment types [9], [10]. Specifically, paper-based check frauds constitute around 55% of the total revenue losses [9]. Likewise, according to a recent American Bankers Association survey, it is claimed that in 2013, check fraud resulted in approximately $645 million of revenue losses in the United States alone [10].

This paper investigates the attack surfaces associated with mobile remote check deposit systems and proposes attack vectors to exploit them. The following observations can be made for *client check truncation systems* (shown in Figure 1):

1) Check deposit functions that used to be performed by trusted and well-guarded entities, such as tellers at banking stations and ATMs for paper-based system, have been delegated to untrusted entities, such as smartphones or scanning and computing systems for the *client check truncation system* (referred to as *the remote check deposit [RCD] system for the rest of the paper*).
2) Replacing the paper-based check with its digital equivalent, that is, a digital image of the paper check has made existing paper-based anti-forgery techniques ineffective.
3) Elimination of a paper trail, as a physical check is not deposited to the bank/financial institution and remains in the hands of the attacker.
4) Advances in digital image processing and machine learning methods have enabled attackers to craft sophisticated digital check forgeries with unprecedented precision.

Based on these observations, a wide array of attacks is proposed to exploit vulnerabilities in the existing remote check deposit system. In this paper, we have also demonstrated the feasibility of implementation and execution of the proposed digital forgery attacks on the existing remote check deposit systems [11].

The proposed attacks are crafted on the client side of the mRCD system by tampering the digital image of the check on the device and software and injecting the forged digital check image into the transaction. To this end, a library of digital image processing modules is developed for image tampering. The image processing library module is developed in the *Matlab*^{TM} programming environment. Shown in Figure 1 is a realization of the proposed attack vectors.

It can be observed from Figure 1 that it is possible to add an intermediate processing step to the normal truncation steps. For example, a digital image of the paper check can be extracted at some point along the path between the digital image analysis and understanding subsystem and the bank-side check-clearing server. An active attacker can digitally alter the digital image using image editing techniques and tools and replace the original image with its tampered version before transmitting it to the check-clearing server. Various attack strategies (e.g., copy-move, block-swapping, cut-and-paste, and so on) can be used to generate a forged copy from either the original digital image of the paper check or from a fake check generation system (as shown in Figure 5). Digitally generating a ''fake'' check from scratch is another attack vector available to the attacker. Details of such an attack vector are provided in Section IV. An attacker can take advantage of AI-based techniques, to generate ''cloned'' digital checks from scratch and inject them into the check clearing system.

To highlight existing vulnerabilities in the mRCD systems and demonstrate the effectiveness of the proposed attack vectors on mobile remote check deposit systems, the proposed attack vectors described in Figure 1 were executed on three online banking Apps for remote check deposit belonging to three Fortune 500 US banks. For the selected online banking Apps, the RCD system was implemented on Android smartphones. In order to mitigate the impact of the breach and to avoid harm to the target bank or customer, the experiments were designed very carefully. In addition, to comply with responsible disclosure practices, the findings of this research were shared with the targeted vulnerable banks more than a year prior to the conference version of this submission [11]. The motivation behind delayed submission of the conference version [11] was to give the targeted vulnerable banks sufficient time to develop and deploy appropriate countermeasures.

A. MAIN CONTRIBUTIONS OF THE PAPER

The major contributions of this paper are:

1) To investigate threat models and attack surfaces for the client check truncation systems being used for mobile remote check deposit systems by almost all major banks in the United States and around the world.
2) To analyze vulnerabilities in mobile remote check deposit systems and possible exploits using advanced digital check forgery attacks.
3) To propose appropriate countermeasures that would thwart such attacks

Specifically, this paper makes the following contributions:

- It highlights vulnerabilities of mRCD systems and the ease with which an attacker can exploit them using either digital check forgery attacks or using ''fake'' check generation using deepfakes (Section III).
- It provides a comprehensive comparison between traditional physical check forgery techniques and modern digital check forgery techniques. In addition, it also highlights the ineffectiveness of traditional anti-forgery techniques developed for paper-based check truncation systems in preventing digital check forgery attacks. (Section II).
- It describes a framework for tampering digital images of a paper check as well as digitally generating ''fake'' checks and depositing them into existing RCD systems (see Section III, IV).
- It outlines an instance of successfully attacking the existing remote check deposit systems of three major banks (in the United States) implemented on Android smartphones (see Sections IV and V).
- It proposes countermeasures to detect digital check forgery attacks on an mRCD system (see Section VI).
- It demonstrates the effectiveness of the proposed image tamper detection-based countermeasures (see Section VI).

The rest of the paper is organized as follows: A brief overview of paper-based check clearing systems and forgery techniques for paper-based check clearing systems and countermeasures to prevent check forgery for paper-based check clearing systems is provided in Section II. Description of the proposed digital check forgery model and its implementation details are provided in Sections III and IV, respectively. Effectiveness of the proposed attack model is outlined in Section V. Detailed descriptions of the possible countermeasures to combat digital check forgeries and their implementation and performance evaluations of the proposed countermeasures are provided in Section VI. Conclusions and future research directions are discussed in Section VII.

II. PAPER-BASED CHECK CLEARING SYSTEM, TAMPERING PAPER-BASED CHECKS, AND COUNTERMEASURES

This section provides a brief overview of the paper-based check deposit and clearing system, common paper check forgery techniques and countermeasures to prevent paper-based check forgeries. It also provides an outline of the evolution of the modern check clearing system, that is, the remote digital check deposit system.

A. PAPER-BASED CHECK CLEARING SYSTEM: AN OVERVIEW

Prior to the 1990s, a check issued by a bank that was deposited into an account of another bank required the physical exchange of the paper check between the two banks for money transfer. To minimize delays associated with paper-based check-clearing process, *central check clearing facilities were developed*. Everyday at these facilities, bank employees used to meet to exchange paper copies of the checks and perform money transfers from payer accounts to payee accounts based on the exchanged checks. To detect forgeries, paper checks being exchanged used to be examined manually by several bank employees responsible for the paper-based check deposit and clearing process chain (e.g., the teller of the receiving bank, often the supervisor of the teller, and employees of the settling bank). The paper-based check deposit and clearing process was labor-intensive, time consuming, costly, and slow.

As check transactions became more common and the volume of exchanged checks increased, magnetic ink-based routing and account numbers enabled machines to read and sort the deposited checks at a much faster speed. However, the clearing process was still dependent on physical exchange of checks at central clearing facilities. The magnetic ink-based check sorting-based check clearing process was still slow.

1) CHECK TRUNCATION

To overcome the aforementioned limitations of the paper-based check clearing system, the US Check 21 Act was approved by United States Congress on October 2004. The US Check 21 Act established the legal equivalence between paper and substitute checks (paper representations of checks with the same information as the original checks), and their electronic representations [12]. The US Check 21 Act aimed to expedite the check clearing process by regulating the pre-existing practice of the *check clearing process* being used by the banks. As a result, older practices of paper-based check clearing could be used together with the newer practice of digital check deposit and clearing.

The next phase of development included the widespread use of *client check truncation systems*. These systems brought check truncation facilities to bank customers via a flood of technologies for remote check deposit and processing. The client check truncation systems include dedicated check scanners, PC clients, and smartphones. This development brought the benefits of electronic check image acquisition and remote deposit to the end customers. This provided valuable savings in the efforts related to physically going to the bank or an ATM terminal and depositing paper-based checks in person. It is important to note that under the newer client check truncation systems, the original paper check remained with the remote check depositing person.

2) SECURITY IMPLICATIONS

It has been noted that fast development of client check truncation systems from a trusted highly controlled environment like bank branches and ATMs to untrusted, unmonitored end users increases the risk of check forgeries. In fact, check truncation on the client side and usage of only an image file in the clearing process have virtually eliminated the deployment of all the anti-forgery techniques developed and perfected over hundreds of years based on paper checks. In addition, due to the recency of this practice, there has been little development of countermeasures in the digital domain that are as effective as those for paper checks. In the following section, we review the anti-forgery advancements of the paper check era, and argue that there are no corresponding equivalent developments for digital checks.

B. TRADITIONAL CHECK FORGERY TECHNIQUES

Traditionally, check forgery used to be performed by physically altering text written/printed on the paper check. The attacker used to alter/tamper various fields on the paper check, including the legal and courtesy amounts, payee names, routing and account numbers, dates, and so on. Forgeries for paper-based checks can be performed in a number of ways, including digitally *photocopying* an original check, altering the digital copy using image processing tools, and printing the forged check using printing devices: *check washing*, where the ink on the original check is erased using chemical compounds and new information is written on the washed check, *check fabrication*, where a completely new check is created; and so on.

Outlined in Table 1 are some common countermeasures being used to combat paper-based check forgeries.[1] The techniques listed in Table 1 aim to make *physical check forgeries* more difficult and to ensure a robust integrity verification process for paper-based check deposit systems. In particular, *Microprint* lines include small type text that appears as a line to the naked eye but can be read when magnified. If a check is photocopied, the microprinted text appears as a solid or dotted line. *Chemically sensitive paper* reacts to common chemicals that may be used to erase the ink on the check by changing color. *Watermarks* are special print patterns visible under light only at certain angles, which may disappear or explicitly appear in photocopied checks. A special case of *watermarks* are *VOID pantographs*, which are hardly visible in the original but become visible in the copy. *UV printing* uses special ink that is invisible to the naked eye but visible under black light (UV). Check scanning devices equipped with UV sensors can detect the amount of the ink and issue warning if the amount is below a certain threshold. *Magnetic ink character recognition (MICR)* also uses a special ink, containing iron oxide for the account and routing numbers, while *bleeding ink* is a special ink that turns reddish when under moisture. As one additional precaution, checks may be printed on special paper that is not available to the general public. The aforementioned countermeasures for paper-based check deposit and clearing

[1] The forgery techniques that can be used on the back-end, such as account reconciliation, have been omitted here as they are common to both paper and digital checks.

TABLE 1. Traditional techniques to combat check forgery.

Countermeasure Technique	Application	Digital Check Deposit
Paper-based	Tampering makes paper artifacts visible	Ineffective
Ink-based	Tampering Ink artifacts are visible if tampered	Ineffective
Print-based	Printed patterns are visible on original check only	Camera-dependent

systems have a considerable success rate (>84%) for forgery detection for paper-based check deposit and clearing systems. It is important to note that the reported success rate of 84% for forgery detection continues to be a widespread problem resulting in huge yearly financial losses [9]. It is important to highlight that the remote check deposit system completely bypasses the countermeasures for paper-based check deposit and clearing systems by removing the very foundation they rely on the physical copy of the paper check and the ink of the printed/written text on it. The print-based countermeasures, which rely on visual properties rather than on chemical and physical attributes, may be potentially used as a protection measure for remote check deposit. This is due to the fact that these countermeasures rely on the consistency of the physical attributes of the deposited checks, which are preserved (to some extent) in the digital check images. These techniques, however, depend on several factors, including camera resolution, camera quality, image quality factor, lighting condition, and pattern quality (if any). Existing digital image forensic methods can be used for integrity verification of the JPEG images deposited for check clearing via the remote check deposit system [13]–[18]. The deployment of existing image forensic tools and techniques is expected to face numerous challenges, such as JPEG quality, lighting condition, etc.

III. DIGITAL CHECK FORGERY ATTACK: *DESCRIPTION*

An attacker can leverage advances in image processing methods and artificial intelligence techniques to craft sophisticated and more powerful attacks on the mobile check deposit system.

A. DIGITAL ATTACK PREMISSES

The proposed attack model assumes that the victim is either an entity that issues a check or a financial institution that receives a check via remote deposit and processes it, whereas, the attacker is an entity with the goal of monetary gain. The attacker can achieve his/her goal either by altering information on the digital image of the paper check or by digitally generating a "fake" check and using the remote check deposit system to inject it into the system for clearing. It is important to mention that this paper does not consider attack vectors consisting of physically altering checks, acquiring digital images of physically altered checks, or depositing digital images of physically tampered checks remotely into the check clearing system. The motivation behind the digital check forgery attack vector is that it is a more low hanging fruit for an attacker than physical check forgery followed by digital image acquisition and injection. This is mainly because, digital check forgery provides numerous advantages over its counterpart of physical check forgery followed by digital acquisition. The attacker is typically the owner of a device and the recipient of the final check that is submitted to the banks. In this paper, we do not consider scenarios where an attacker has some degree of control over the device of an unaware user and uses that device to perform the attacks. Even though possible, we believe these scenarios to be less likely to occur. In fact, as will be explained later in the paper, the attack relies on modification of system software, which is signed, and on the login credentials of the banking applications.

B. DIGITAL CHECK FORGERY ADVANTAGES

Digital check forgery attack has the following advantages over the traditional paper-based check forgeries:

1) **Precision and Flexibility**: Available digital image processing methods and recent advances in image manipulation tools provide an attacker with the opportunity to manipulate a digital check image at the pixel level. The available image processing and manipulation tools provide an attacker ability, flexibility, and precision unrivaled by the physical check forgery techniques. For instance, consider the amount area on a physical check (as shown in Figure 2) and its digital equivalent acquired by a ($5MPixel$) digital camera. The physical area measures approximately $3 \times 0.8\ cm^2$ on the physical check, whereas, the corresponding digital image measures $296 \times 87 = 25,752$ pixels. In addition, modern digital cameras provide a bit depth of 24 bits (8 bits per color channel, i.e., red, green, and blue) per pixel. The available digital image processing and manipulation tools provide an attacker 16.8 million (i.e., 2^{24}) color options to manipulate the target pixel(s). From a practical stand-point, an attacker can choose from a smaller set of colors associated with darker ones for the amount to show; however, the selected subset is still large enough to introduce a wide range

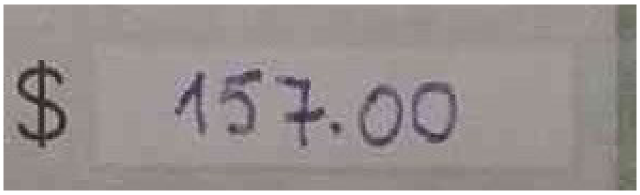


FIGURE 2. The amount area of a check (3 times magnified).

of modifications. To reach a similar precision level in a physical check, an attacker needs to select and manipulate a region equal to $0.93*10^{-4}mm^2 = 93\mu^2$. With the advances in scanning technology and the availability of higher-resolution digital cameras, digital forgery is expected to be even more precise and undetectable. Moreover, digital check tampering does not introduce any physical damage to the paper check; therefore, it is reusable for crafting other attacks. In particular, even though an attacker with a physical tampering attack vector may not need the level of precision available in the digital domain, physical forgery is expected to trigger countermeasures such as ink bleeding and chemically sensitive paper to make the paper check unusable.

2) **Unlimited Trial and Error**: The process of digital check forgery is fundamentally different from the process of physical check forgery in several ways. Firstly, tampering operations in the digital domain provide an attacker an unlimited capability of "*undo*" therefore always ensuring an original state of the digital image. Likewise, an attacker can rely on a copy of the original for tampering, which enables the attacker to play with an unlimited number of digital images to perfect the forgery before submitting the check to the bank, thus minimizing the risk of detection. On the other hand, tampering a physical check cannot be attempted more than once or twice without damaging it. Secondly, the forgery detection techniques for paper-based checks such as special paper, special patterns, ink bleeding, and so on are ineffective for digital check forgery attacks.

3) **Absence of Trails for Forgery Detection**: The traditional check clearing systems, such as in-person check deposit to a teller at a banking facility or the ATM. keep a paper trail that is used for forgery detection either in real time (in the case of a traditional check transaction) or during a post-clearance audit. The remote check deposit and clearing systems do not leave any paper trail with the financial institution. It is therefore not possible to use existing countermeasures for a paper-based check clearing system for forgery detection for a remote check deposit and clearing system.

4) **Use of Untrusted Client Check Truncation Systems**: Shown in Figure 3 is an illustration of the trust models for three check clearing systems: teller-based, ATM-based, and remote check deposit and clearing systems. It can be observed from Figure 3 that the remote check deposit and clearing system has shifted the trust level from trusted, tamper resistant, and well-protected entities (e.g., teller centers or ATMs under video surveillance) to a set of untrusted (vulnerable to tampering) entities under an attacker's control. It can also be observed from Figure 3 that for a remote check deposit and clearing system in which the image acquisition and image processing analysis system is under an attacker's control, a determined attacker can interpose at any point along the path from the scanning device to the network device that to manipulate the acquired digital image and inject forged images into the clearing system.

5) **Reduced Risk of Punishment**: In addition to avoiding classic anti-forgery techniques and reducing the risk of detection, by not requiring the users' physical presence, remote check deposit reduces the risk of capture for an attacker as well. Furthermore, if the forgery is detected by some other means, an attacker's denial is more plausible, since law enforcement agencies have to prove beyond reasonable doubt that the attack was actually carried out by the attacker. In case of perceived risk, an attacker may get rid of their smartphone or any other *client truncation system* or claim they were stolen, thus complicating even more the proof by law enforcement agencies.

6) **Facilitation of Traditional (Physical) Forgeries**: Another venue of attack enabled by remote check transactions is the physical modification of a check followed by submission of the physically forged check using a check truncation system. We believe that this type of attack is facilitated by the transition to remote check transaction systems, since it reduces the risk for the attacker, while making it more difficult to detect forgeries without the physical paper. However, this type of attack is less general than the digital forgery attack, since in certain cases the attacker has to defeat potential physical countermeasures embedded in the checks. For example, attacks specializing in check washing would probably fail on checks printed on chemically sensitive paper, thus damaging the only available copy, while no chemical properties are involved in digital forgery. We strongly believe that the fact of having unlimited digital copies and the power to manipulate single pixels in JPEG images renders digital forgeries more desirable for attackers. In general, however, remote check truncation systems facilitate this kind of attack as well.

To summarize the availability of *easy-to-use*, *powerful*, *sophisticated* digital image processing tools; the availability of unlimited trial and error for tampering; the elimination of the *paper trail*; and the use of untrusted *client check truncation systems* are enablers for a powerful digital check forgery attack.

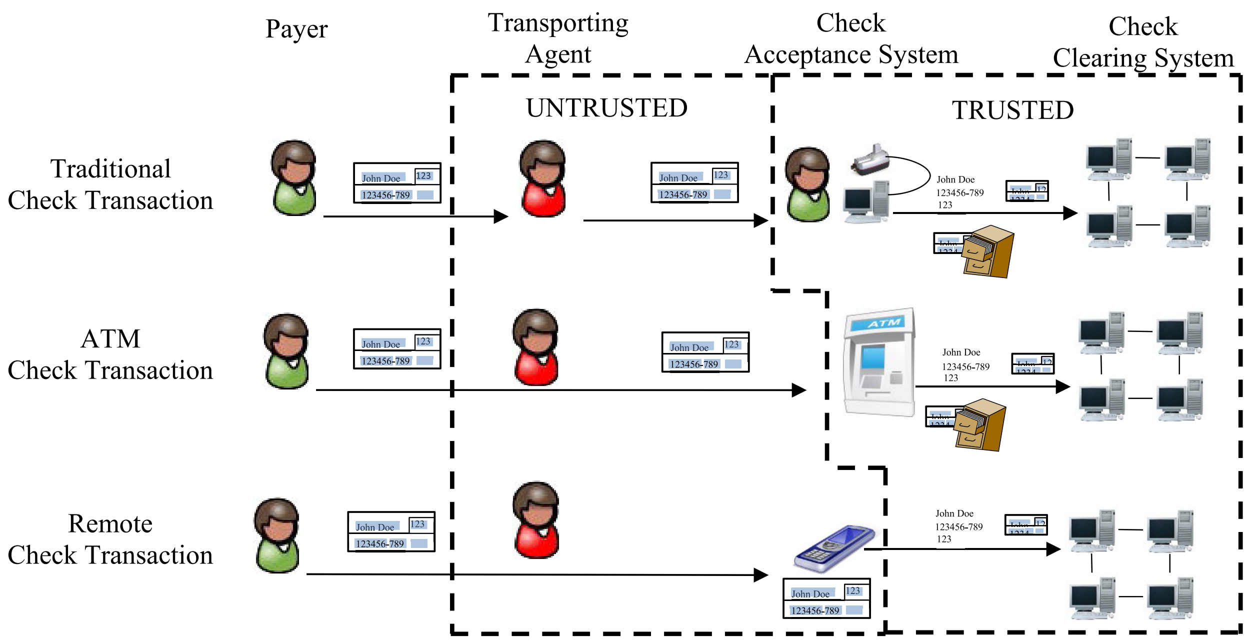


FIGURE 3. Illustration of trust models for Teller, ATM and remote check deposit and clearing systems.

C. AN IMAGE PROCESSING FRAMEWORK FOR DIGITAL CHECK FORGERIES

We assume that an attacker does not have any prior knowledge about countermeasures that may be deployed at the check-clearing server. This means that the attacker must create a forged image that can defeat any of the existing forgery detection mechanisms now in place. The goal of the attacker is therefore to generate a forged image such that distortion due to forgery is below the detectable range or space.

The attacker can adopt a forgery strategy that maximizes the attacker's chances to succeed. The check clearing system, on the other hand, can also adopt a strategy that maximizes its chances of detecting forgeries in the deposited checks. To this end, the check clearing system can exploit the fact that almost all image manipulation methods do leave characteristic artifacts in the resulting forged image. For instance, block boundary artifacts are the most common image tampering artifacts introduced in tampered images. Boundary smoothing or blurring is generally applied along the border(s) of the modified region(s) to mitigate the effect of irregularities [19]–[23]. Subsequent filtering based on localized average filtering, boundary diffusion filtering, etc., can be used to mitigate block-boundary artifacts. In addition, other advanced image processing methods such as super-resolution image processing can also be used to mitigate tampering artifacts. It is important to highlight that super-resolution imaging-based digital image tampering may not fit in the commonly known image tampering methods such as copy-move, splicing, etc.

It is also assumed that the attacker is aware of existing image tamper detection techniques [24]. These techniques rely on the fact that almost all image manipulation models do leave characteristic artifacts in the resulting image, and use underlying artifacts for forgery detection. These artifacts may be detected by several passive detection techniques such as bispectral analysis, JPEG compression-based methods, etc [19]–[23]. To avoid forgery detection, the attacker can use sophisticated manipulation methods, e.g., deepfakes. Details of the proposed next generation of digital check forgery methods are outlined in § III-D.

D. ATTACK DESCRIPTION: DIGITAL IMAGE PROCESSING

To generate a tampered image of the check, an attacker can tamper with one or more of the fields in the authentic image of the check, ranging from (i) *computer-generated fonts* (e.g., account #, routing #, etc.), (ii) *handwritten or typed text* (in the payee name and the courtesy-amount and legal-amount areas), or (iii) *the payer's signature*. Let $I_a^{(i)}$ and $I_t^{(i)}$ denote authentic and digitally tampered/forged images of the check scan respectively, and $i = \{1, 2, \ldots, m\}$ denote the image blocks comprising the minimum forge-able information. Mathematically, the relationship between $I_a^{(i)}$ and $I_t^{(i)}$ can be expressed as:

$$I_t^{(i)} = \mathbf{\Phi}\{I_a^{(i)}\} \tag{1}$$

where $\mathbf{\Phi}\{\cdot\}$ represents the tampering process.

A wide range of digital image tampering techniques that are available to the attacker to generate a tampered image from the original ranging from simple techniques such as copy-move, cut-and-paste (or splicing), seam carving, etc. [20] to sophisticated techniques that can be customized to fit the content can be modeled using $\mathbf{\Phi}\{\cdot\}$. The overall goal of the digital check forgery is to keep the value of the function $J < \rho$, where ρ is the threshold and J can be computed as:

$$J = \frac{1}{m}\sum_{i=1}^{m}\left(I_a^{(i)} - I_t^{(i)}\right)^2 \tag{2}$$

An attacker can take advantage of available advanced digital image processing and manipulation methods to achieve his/her objectives. To this end, an attacker can use the authentic image to construct a dictionary consisting of computer generated fonts, handwriting, and signature templates. Depending on the completeness of the attackers' dictionary, (s)he can then decide how many fields to tamper with to achieve the desired financial gain. Consider, for example, Figure 2 where an attacker wants to target the courtesy amount area 157, (s)he has many options available such as $\{175, 517, 571, 715, 751\}$. The selection of a possible modification can be modeled as a **min-max problem**, where an attacker can choose a modification strategy that results in minimum distortion/artifacts in the forged image while maximizing the financial gain.

To solve this min-max problem, an attacker can use either *copy-move* or *block-swapping* or both image manipulation methods to tamper with the legal and courtesy-amount fields. Next, we briefly discuss commonly used image tampering techniques:

- **Copy-Move Forgery (CMF):** In the case of CMF attack, an attacker can select a block of interest (of an arbitrary size and shape), $I_a^{(s_j)}$, from original image $I_a^{(i)}$. The source block $I_a^{(s_j)}$ is used to replace the target block $I_a^{(t_j)}$, in the original image, by applying a series of image transformation operations such as bilinear transformation on the $I_a^{(s_j)}$. This block-replacing process can be repeated until a desired tampered image is obtained. The copy-move-based methods are useful when the attacker wants to tamper with information in an original check image. The copy-move-based tampering methods are robust to countermeasures that rely on inconsistencies due to lighting conditions or camera noise. The copy-move technique is one of the most common image tampering methods used today [19], [20]. The reasonable amount of research activity on this topic [19], [20] confirms this observation.
- **Block-Swapping:** In the case of block-swapping attack, an attacker selects a pair of blocks (of an arbitrary size and shape), $I_a^{(s_j)}$ and $I_a^{(t_j)}$, and swaps them. This block-swapping process can be repeated until a desired tampered image is obtained. In case of block swapping, image tamper detection methods that rely on duplicate regions in the test image will be unable to detect block-swapping based forgeries.
- **Cut-and-Paste (or Splicing):** In **splicing**-based tampering, an attacker is assumed to have access to a set of n images of checks, which he uses to generate $I_t^{(i)}$. The splicing attack is probably more common than copy-move and block-swapping attacks, because it allows creation of images with a very different content. The extensive research activity on this topic [19], [20] confirms this claim.
- **Advanced Image Processing and Machine Learning Based Methods:** What if an attacker has access to multiple copies of the checks issued by the same payer? The attacker can take advantage of advanced image processing, and machine learning methods to create the most powerful attack. More specifically, let us assume that an attacker has access to multiple checks (e.g. a set of K sample images issued by the same payer); an attacker then can take advantage of image inpainting methods for check washing; and apply sequential models such as recurrent neural networks (RNN), long-term short-term memory (LTSTM), etc. to learn payer's handwriting; to generate a "fake" check-image, I_t. Shown in Figure 4 is a conceptual block diagram of digital check washing and forged check generation using advanced image processing, and machine learning methods for synthetic check-image generation.

An attacker can take advantage of these attacks to alter the digital image of the check. Specifically, the attacker can use the aforementioned attacks to tamper with various fields in the digital image of the check, including *payee name*, *courtesy amount*, *legal amount*, signature, *date*, *check number*, and *routing number*. For instance, **legal- and courtesy-amount field** alterations can be achieved using cut-and-paste, copy-move, or block swapping attacks. Legal- and courtesy-amount alteration also requires a moderate-to-advanced image processing skill level and a moderate-to-complex difficulty level.

Similarly, the **payee name field** can also be tampered with. Here an attacker can use a dictionary consisting of handwriting templates to insert a desired name. To achieve this goal, an attacker can take advantage of common image tampering methods, e.g., copy-move, block-swapping, cut-and-paste, etc., to tamper with the *payee* name field. For example, in the case of copy-move based tampering, the attacker first erodes the *payee* name and obtains a forged check by printing the desired payee name using templates from the attackers' dictionary.

An attacker can also tamper with **computer generated fonts** in the authentic image of the check, that is, **check #**, **count #**, **routing #**, etc. For such forgeries, an attacker can use any one the available simple image tampering techniques to obtain a forged check. This is a relatively simple attack compared with the payee name or legal- and courtesy-amount attack. To generate a forged check by altering the computer-generated fonts, an attacker first decides how many numerals need to be modified in the resulting forged check, then selects a template from the dictionary containing a numeral from the required combination and replaces it, and repeats this process until a desired check number sequence is obtained.

To tamper with the **payers' signature**, the attacker can take advantage of advanced image processing techniques such as image morphology to extract a signature template from the original check(s). Once the attacker has an access to the payer's signature, any of the image tampering techniques discussed above may be used to obtain a forged check. This forgery is very hard to detect using the existing image tamper

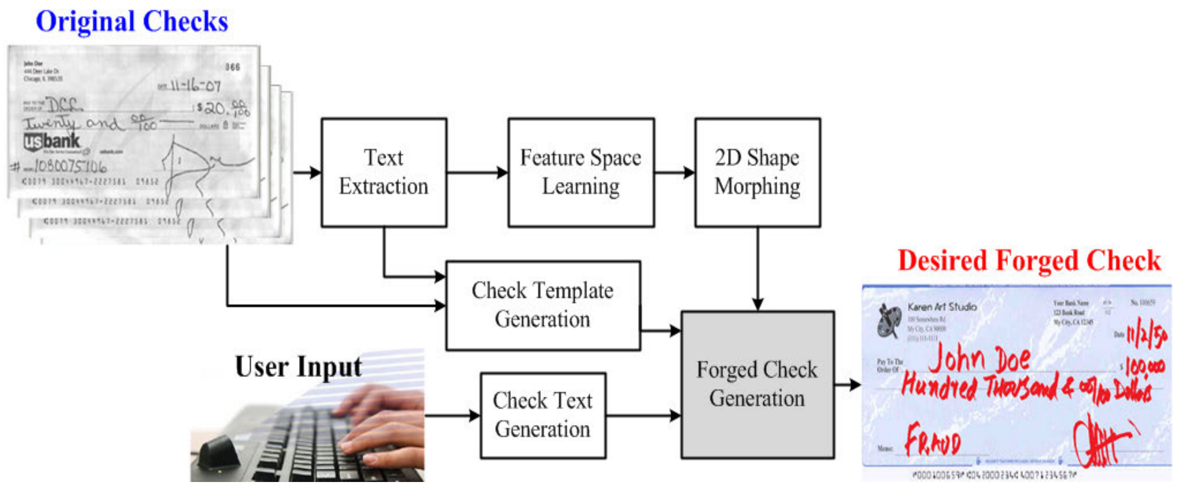


FIGURE 4. A conceptual block diagram of digital check washing and forged check generation using advanced image processing, machine learning, and computer graphic methods.

detection methods [19], [20]. The payer signature tampering would require a moderate-to-complex image processing skill level and a moderate-to-complex difficulty level. Moreover, the attacker can also use synthetic signature generation techniques to automatically render the payer's signature.

- **Signature/Handwriting Synthesis:** Synthetic signature generation techniques model the human's cognitive and neuromotor behaviors (i.e. the neuromuscular process) to imitate the handwriting [25]. Through signature image analysis, these methods also perform the mapping of several signature components i.e. pen ups, short and long strokes, pen velocities, character variability, tilt, and force ranges to effectively generate the pen traces [26]. To achieve this, signature morphology is performed as a first step to capture the connected and disconnected components of the signature. Afterwards the letter representation and connectivity is defined by overlaying hexagonal grids over the signature. The definition also reveals the stroke profile and pen up information through the non-connecting letters and word endings. Afterwards, the impulse finite filters, i.e. Kaiser filters, are applied and analyzed through probability density functions to obtain the trajectory's velocity information and generate the static image of the signature by further applying ink-deposition models [25].
  A similar approach can be used for handwriting synthesis due to the fact that handwriting and signatures have overlapping attributes.
- **Attack on Signature Verification:** The digital check verification system considers the signature verification as a standalone activity and in contrast to other payee information extraction exempts the signatures from the OCR processing that is merely attributed to the variable representation of the letters. The signature verification process usually comprises the following steps: preprocessing, feature extraction, and classification of the signatures through machine learning-based models as forged or original [27]. However, the attacker can delude the check verification system into rejecting the original signatures and accepting the forged one. For this, the attacker can attack any component of the payers' signature verification pipeline: [28]. For instance, the attack can be induced at the preprocessing phase, which comprises the following steps: signature extraction, noise removal, size normalization, signature representation, and alignment. To reject the original checks, the attacker can complicate the signature background by fusing it with complex textures hence, making the signature extraction difficult or incorrect. The incorrect signature extraction results in incorrect feature extraction, and consequently the classification system will misclassify. The purpose of this attack could be to borrow more time to ensure that the forged check occurs prior to that of the original check.
- **Attack by Overcoming the Fake-Signature Vulnerabilities:** The synthetic signature generation techniques that consider neuromotor behaviors e.g. pen ups, variation in pen velocities during short and long strokes etc., generate forged signatures that are difficult to detect. In contrast, human counterfeiters are unable to perfectly apply all these principles, and therefore the probability is high that the signature verification system will detect those signatures as forged. The primary drawbacks occur in the form of the non-smooth trajectories due to tremor, deviation of the salient points, and high variance of the stroke velocities. However, the existing vulnerabilities in manually forged signatures can also be addressed by applying neuromotor principles to launch an attack that has a high probability of success. For instance, the tremor that results in non-smooth trajectories can be overcome through linear interpolation and by applying a low-pass filter, i.e., the FIR filter [28]. The salient points of the original signatures can be estimated through a salient-point estimator [29] that defines the curvature of each salient point at different scales and returns a curvature matrix. Afterward, the salient points of the

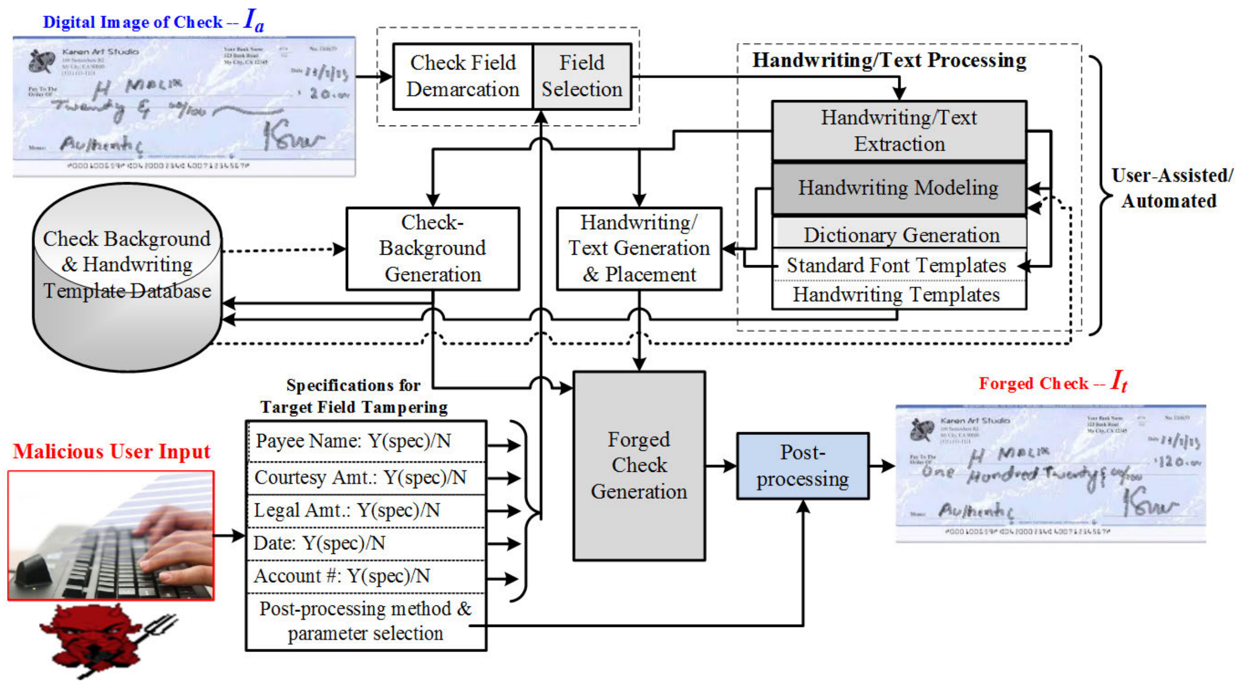


FIGURE 5. Block diagram of the proposed digital check washing and forged check generation framework.

forged trajectory are computed and normalized through the curvature matrix of the original signatures [28], which consequently returns a velocity profile the same as of the genuine signature. The reconstructed signatures will now have a higher probability to deceive the check verification system.

- **Digital Check Washing:** Digital check images are vulnerable to the check washing attack. When a check washing attack is applied on the desired fields e.g. payee name, legal and courtesy amount etc., it generates the empty fields by removing only the hand-written text that can be regenerated through handwriting synthesis in terms of the payer's handwriting. For digital check washing, image inpainting methods e.g., [30], [31], can be applied. However, the major limitation of the image inpaiting methods is that they demand that the masked information of the regions be washed out, whereas in real scenarios, this information is not always available. However, through deep learning based image inpainting methods e.g., [32], this limitation can be very easily addressed. For instance, the CNN models proposed for segmentation e.g., [33], can be trained on the ground truth inpainted images with ground truth text information available to segment the regions of interest i.e., the handwritten text that needs to be washed out [32]. The CNN returns the output in the form of the region proposals that can be used as masks required by the image inpainting methods. The image inpainting methods employ region-filling approaches by acquiring the intensity information from neighboring pixels [34].

  Once the digital checks are washed out, the new information can be generated by applying the handwriting synthesis approaches discussed earlier.

As digitally generated forged images do not leave any traces in the forged image, this forgery therefore is very hard to detect using the existing image tamper detection methods [19], [20]. Shown in Figure 5 is the proposed framework that enables the attacker to achieve his/her objectives, that is, tamper the desired region in the digital image of the check and avoid detection. The proposed forged check generation system enables an attacker to systematically alter the desired field(s) of interest, including *payee name*, *courtesy amount*, *legal amount*, *date*, *check number*, and *routing number*. The content of the target field(s) can be either handwritten or printed text. Moreover, the proposed forged check generation system also enables an attacker to generate a forged check from scratch. The next section provides a brief description of the proposed forged check generation system and outlines details of various attack vectors that can be implemented using the proposed system.

E. DETAILED DESCRIPTION OF THE FORGED CHECK GENERATION SYSTEM

The input to the proposed forged check generation framework is a digital image, I_a, (acquired using an imaging device, e.g., a digital camera) of the original paper check and attacker provided attack vector specifications. These specifications include the target fields to be altered, the type of alterations, the postprocessing method (to hide traces of modification), and so on. For example, the attacker can target various check fields including the courtesy amount of "*30.00*" (in the courtesy amount field) and the Legal amount of "*Thirty & 00/100*" (in the legal amount field) in the original input image, I_a. To achieve this goal, the attacker specification will include altering Courtesy Amount from "*30.00*" to "*130.00*" and likewise, legal amount field alteration from "*Thirty & 00/100*" to "*One Hundred Thirty & 00/100.*" The output of the forged check generation system is a digitally tampered/altered image, I_t.

- **Field Demarcation and Field Selection (FDFS) Unit**: The FDFS unit processes the input image, I_a, and demarcates the boundaries of the target check fields. The digital image of the check and the user/attacker provided specifications are inputs to this unit, whereas the selected fields are the outputs. It allows the user to enter target field specifications through a graphical user interface. This unit is also capable of automatically detecting field demarcations, which is similar to the automatic check reading system outlined in [35].
- **Handwriting/Text Processing (HTP) Unit**: The selected fields from the FDFS unit are applied as input to the HTP unit. The main task of this unit is to analyze the selected check field(s) for text extraction, modeling of handwriting style, and dictionary construction from handwriting and standard font templates. The HTP unit can be further divided into three processing subunits: (1) handwriting/text extraction subunit, (2) handwriting modeling subunit, and (3) dictionary construction subunit. Details of these three subunits are provided in the following:
    1) **Handwriting/Text Extraction (HTE) Subunit**: The output of FDFS unit and the attacker specification are input into this subunit. It analyzes the selected fields for text extraction. Specifically, for text extraction it relies on methods based on digital image morphology [36]. For example, a series of *dilation* and *erosion* operations along with user/attacker feedback are used for handwriting/text extraction. Extracted text or handwriting is the output of this subunit.
    2) **Handwriting Modeling (HM) Subunit**: This subunit is used for handwritten checks. If the check is handwritten then the output of the HTE subunit is applied as input to this subunit. For handwritten fields such as the payee name, date, courtesy amount, and legal amount fields that provide useful information related to handwriting style, learning the handwriting style can help an attacker to bypass an eventual handwriting verification system. To this end, this subunit aims to model the handwriting extracted from the input check image using active shape modeling as discussed in [37]. The output of this subunit is a handwriting model that can be use to print fake check with victim's handwriting.
    3) **Dictionary Construction (DC) Subunit**: The output of the HTE unit is also applied at the input of this subunit. It processes the extracted text/handwriting from the selected fields to extract a template for each character. The motivation behind selecting character templates is using them for forged or fake check generation. Specifically, a series of image processing operations—dynamic segmentation, slant correction, size normalization, and thickness normalization are used for template generation [35]. The character templates used for dictionary construction for each victim and check type are stored in a database that can be used for forged or fake check generation.
- **Check-Background Generation (CBG) Unit**: Selected fields are applied at the input of this unit. It uses segments of the selected field and a digital copy of the input check to estimate the background of the check. Specifically, it "**digitally washes the input check**" by interpolating the pixels corresponding to the extracted text and filling them with values similar to the surrounding background. The check background generation task can be executed with varying levels of sophistication, for instance, by using background check images stored in the database and using advanced image processing methods to match the **washed pixels** in the target field area(s) with the check background [38]–[40]. The recent advances in artificial intelligence and super-resolution image processing methods are expected to provide attackers even more power to execute such tasks.
- **Text Generation and Placement (TGP) Unit**: The main functions of the TGP unit include generation and placement of new text in the target fields of the tampered check. Outputs of the HTE, HM, and DC subunits are applied at the input of this unit. It can be used to generate and place new text. Specifically, the TGP unit will rely on the DC subunit, which contains templates of victim's checks, for writing text/handwriting in the target fields using template-based active shape modeling as discussed in [37]. The goal of this unit is to ensure consistency of the handwriting and fonts in the tampered check with the original check. Moreover, image-processing operations, including resizing, rotation, and spacing can also be used to achieve goals of the TGP unit.
- **Check Background Templates and Character Database (CBTCD) Unit**: The CBTCD stores the estimated check background templates, check-issuer specific handwriting styles, models, and handwriting word- and character-level dictionaries. During the proposed forged check generation processing, the TGP and FCG units request that the CBTCD unit provide information not readily available from the input image of the check, such as character templates and check backgrounds.

TABLE 2. Forgeries and their realizations using the proposed forged check generation system.

Forgery Type	Processing Units/Subunits Involved
Date	FDFS → HTP(HTE & DC)→ TGP → CBG → FCG
Legal- & Courtesy-amount	FDFS → HTP (HTE, HM, DC)→ TGP → CBG → FCG
Payee Name	FDFS → HTP (HTE, HM, DC)→ TGP → CBG → FCG
Signature	FDFS → HTP (HTE & DC)→ TGP → CBG → FCG
Check #	FDFS → HTP(HTE & DC)→ TGP → CBG → FCG
Fake Check Generation	HTP (DC)→ TGP → CBG → FCG

- **Tampered Check Generation and Post-processing (FCG) Unit**: The task of the FCG unit is to suppress tampering artifacts, e.g., text or field boundary imperfections. The FCG unit enables the attacker to select one or more post-processing operations from the available options. It is worth highlighting that some post-processing operations, such as linear filtering and nonlinear smoothing do leave statistical artifacts [24] in the resulting forged check image. To get around such issues, an attacker can take advantage of anti-forensics methods as discussed in [24].

1) Attack Vector(S) Implementation Using the Proposed Forged Check Generation Framework: The proposed framework enables an attacker to craft a wide array of attack vectors, ranging from simple modifications in the target field to sophisticated tampering, including generating a fake check based on attacker inputs. The proposed framework in Figure 5 enables an attacker to realize all known and future attacks on digital checks. Specifically, the proposed forged check generation system (shown in Figure 5) enables an attacker to alter one or more fields in the digital image of the check.

Moreover, with the database of check backgrounds, character templates, and learned models of handwriting, which can be populated over time, coupled with advances in artificial intelligence methods, computer graphics tools, and post-processing techniques that aim to bypass counter-forensic functionalities, more sophisticated forgery attacks are possible. It is import to highlight that the proposed proposed forged check generation system (shown in Figure 5) is fully capable of *generating a fake check from scratch*. Generating "fake" checks without using an actual paper check is not science fiction anymore.[2] Listed in Table 2 are some common forgeries/attack vectors targeting one or more of the check fields including *payee name*, *courtesy amount*, *legal amount*, *date*, *check number*, *routing number*, or generating a fake check and the realization of these attack vectors using the proposed forged check generation system.

[2]**The Reality-Distorting Tools Of The Future** - URL: https://www.sciencefriday.com/segments/the-reality-distorting-tools-of-the-future/

IV. DIGITAL CHECK FORGERY ATTACK: *IMPLEMENTATION*

This section provides an overview of the implementation of an instance of the proposed forged check generation system for three *client remote check deposit systems* for the Android platform. The implementation of each instance requires library instrumentation to provide access to a digital image of the check acquired by the mobile remote check deposit system for modification and a library for digital check tampering. The instrumented library also enables the attacker to inject a tampered check to the server for clearance. Details of both the library instrumentation and digital check tampering are provided next.

A. LIBRARY INSTRUMENTATION

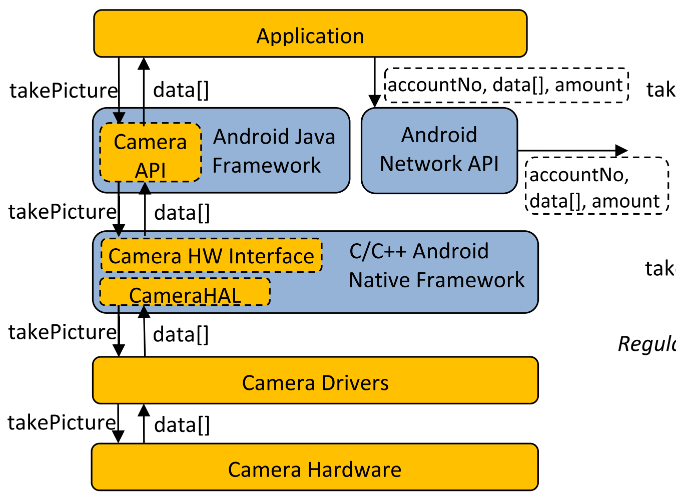
The aim of library instrumentation is to achieve a transparent interposition between the point of the digital check image acquisition and the point where digital check image is sent over the network to a financial institution for check clearance. The instrumentation described here is Android-specific, but similar instrumentation can be applied to other implementations of *mobile remote check deposit and clearing systems*. More specifically, it includes:

1) Software modification with the aim to analyze the communications between the different application components
2) Identification of the interposition points to extract original digital images and inject forged images
3) Implementation of the original digital image extraction and injection of forged check operations

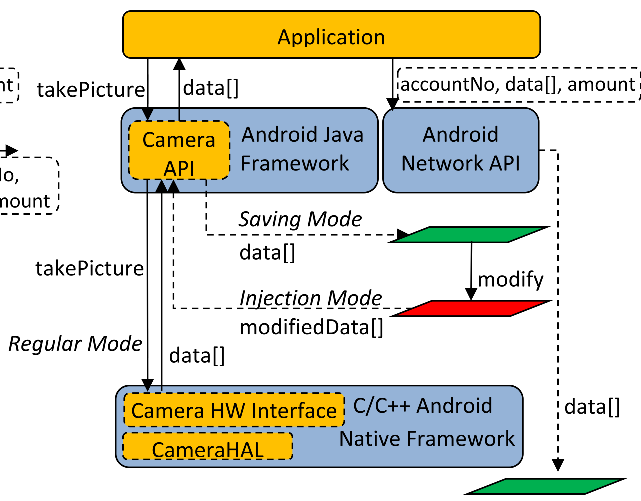
It is important to highlight at this point that the bank applications are deliberately treated as black boxes due to the following reasons: First, we wanted to demonstrate the generality of the proposed attack vector and make it application independent. Second, the end user licence agreements (EULAs) of those applications specifically prohibit decompilation or modification of the binary or source code.

Shown in Figure 6.a is the original software stack for the camera subsystem. It is important to highlight that the *mobile remote check deposit and clearing system* in Android lies entirely inside the device and includes the full software and hardware stack from the camera hardware to the bank application. It can be noted from Figure 6.a that the bank applications rely on the camera and network APIs during a check transaction.

In Android, the camera subsystem is implemented in the Java `android.hardware.camera` package and related C/C++ code residing in the lower layers, while the network APIs are implemented by several libraries, among which is the Java Apache HttpClient library. To capture the operations during a check deposit transaction, we introduced DEBUG output messages at several key points inside these libraries.



(a) Original Software Stack



(b) Modified Software Stack

FIGURE 6. The original and modified camera subsystems.

Each message prints out the line of the source code that is being executed as well as the stack trace at that point. The objective of these instrumentations is to gain a clear understanding of how these libraries interact with the applications at each step of the check deposit transaction.

To acquire a digital image of the paper check, an application issues a request to the class `android.hardware.camera.Camera`. Inside this class, another private class is responsible for handling callbacks and ultimately forwarding the (JPEG) image data to the application inside an array of bytes. Next, the application processes the image and sends it to the network APIs to be delivered to the bank servers. Further instrumentation of the Camera and HttpClient classes allowed us to extract the original images being delivered to the bank applications and the processed images being sent over the network.

This analysis suggests following two alternatives for the altered digital image injection point:

1) In the camera subsystem before the image is received and processed by the application
2) In the network subsystem, after the image is received and processed by the application and before it is encrypted and sent over the network

The latter alternative, however, poses a greater risk, since it may interfere with eventual image processing inside the application. In addition, not all applications use the Apache HttpClient library. Therefore, we chose to instrument the camera subsystem for injecting the forged image. The resulting system is depicted in Figure 6.b using dashed arrows.

The proposed instrumentation provides following three different modes of operation for the Camera subsystem:

1) **Image saving mode**, where a copy of the image data is saved as a JPEG file on local storage and the image data are forwarded to the application,
2) **Image injection mode**, where the image data are retrieved from a file on local storage rather than from the underlying layers,
3) **Regular mode**, which is the original mode of the camera subsystem, where the image data are forwarded to the applications that use the camera.

These modes can be enabled/disabled using a simple configuration file saved on the local storage of the smartphone.

We seek to minimize the impact of the instrumentation, that is, to introduce as few disturbances as possible into the data received by those applications, which is achieved by not interfering with the applications' operations. For instance, since the applications request JPEG data rather than RAW data, it was therefore decided not to change the option to RAW. In fact, even though the RAW data returned by the camera may provide the original dataset on which to perform the forgery, a subsequent JPEG compression is still needed to pass the modified image to the application. It is important to know quantization parameters, as these parameters are used for compression by the camera, and the subsequent compression (done by the proposed framework) may be different from that performed by the camera, thus potentially disturbing the data.

B. DIGITAL CHECK MODIFICATION

For proof-of-concept implementation of the proposed attack model, a simple digital check forgery (due to the sensitivity of the experiments) was introduced by tampering only the legal- and courtesy-amount fields of the digital check image. This digital check forgery was implemented in the MATLAB environment. Our implementation of the framework units *FDFS*, *HTP*, *TGP*, and *FCG* is described in § III-D. To assist the attacker in interacting with subunits of the proposed forged check generation system, e.g., *FDFS*, *HTE*, and *CBG* units, a graphical user interface (GUI) was also developed. The GUI visualizes the check and allows the user to provide an input vector consisting of the locations of the target fields and the post-processing method to be used along with its parameters.

More specifically, starting from the original check (Figure 7.a), the user-input-driven *FDFS* unit selects the

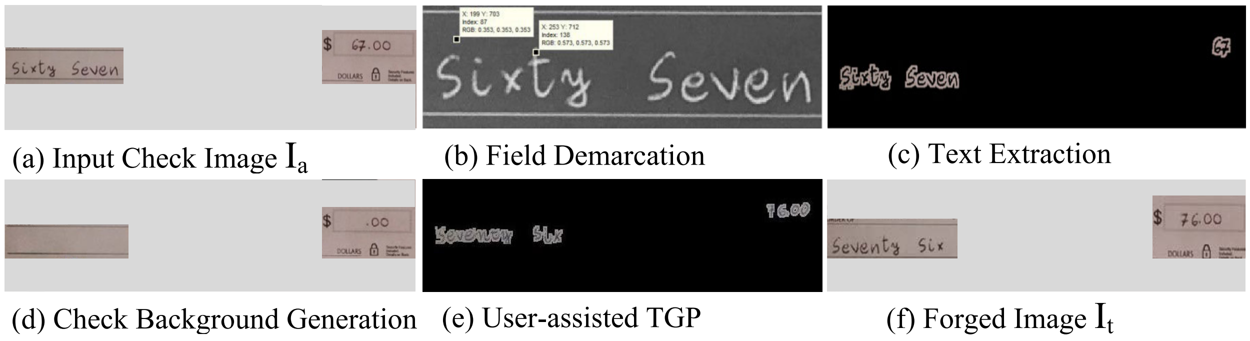


FIGURE 7. Illustration of forged check generation.

two fields (e.g., legal- and courtesy-amount fields shown in Figure 7.b). The output of the FDFS unit is applied to the input of the *HTP* unit, which applies background subtraction and relative thresholding to identify the handwritten text in those fields (Figure 7.c). The developed GU for the HTE subunit allows the user to select portions of the target field(s), decompose the field text into individual characters and use them to build a character dictionary of the text in the selected fields. Next, for each targeted field, the user-assisted *TGP* unit sequentially selects the desired set of characters from the dictionary and places them in the selected field (see Figure 7.e).

To achieve check tampering, each targeted check field is selected, analyzed and processed by the *CBG* unit, which "**digitally washes**" the selected field by replacing the original text/handwriting with the desired text/handwriting (e.g., Figure 7.d). The *FCG* unit then merges the background image with the output of the TGP unit to obtain the final image (Figure 7.f). The post-processing unit processes the tampered check with the objective of minimizing artifacts resulting due to the image modification. An averaging filter of 3×3 pixels is used to obtain the tampered check image after post-processing. Finally, the post-processing step uses Exiftool [41] to copy and update as necessary the JPEG Exif metadata from the original file into the forged check image. For instance, the creation timestamp is modified to coincide with the injection time rather than with the capture time.

V. EXPERIMENTAL SETTING AND PERFORMANCE EVALUATION

1) EXPERIMENTAL SETUP

A Samsung Galaxy Nexus smartphone with Android 4.1.2 (Jelly Bean) OS running on it was used for experiments. We highlight at this point that, since the software running on Android devices is the same for all the devices, this attack will work with other devices as well. In fact, in preliminary experiments with another device (Samsung Galaxy S2), we were able to replicate the different steps of the attack, stopping at the final submission.

The Android source files were downloaded from the official Google Android repos [42], and those implementing the camera and network APIs were modified as described in Section IV-A. Finally, to gain root access a *userdebug* build was flashed into the target smartphone.

To demonstrate the effectiveness of the proposed attack methodology, three major Android banking applications were selected. The selected applications are being used by millions of people. Selection of these applications was partly motivated by the sensitive nature of the attacks, which involved using our own accounts in those banks, and by the availability of the remote check transactions functionality in the Apps. In fact, not every banking application offered this functionality at the time of the experiments.

2) BANKING APPLICATIONS DESCRIPTION

After a user logs in, each application presents a screen with the check and instructs the user to take pictures of its front and back. Next, the user is required to select the account # where the check must be deposited, type in the amount, and finally submit the check. Up to the final submission step, the transaction may be canceled at any time by the user. Using the network APIs, the application transmits the two images and the data submitted by the user to the server. On the server side, optical character recognition (OCR) software is used on the check's areas of interest and a confirmation message is sent back to the user.

3) PRELIMINARY EXPERIMENTS AND ANALYSIS

Before launching the actual attacks, several preliminary experiments were performed to gain an understanding of:

1) The properties of the images captured by the camera and of those sent to the server as well as applications' features
2) The server-side operations, in particular tolerance to errors, picture quality, OCR capabilities, and possible human involvement in the clearing process

4) IMAGE PROPERTIES AND APPLICATION FEATURES

Using the instrumented libraries, we initiated several transactions with untampered checks, most of which were intentionally aborted before the final submission.

The results of these experiments are outlined in Table 3. The first column of Table 3 represents the bank; the second represents the JPEG quality, approximate file size, dimensions, and the metadata format of the images (Exif or JFIF)

TABLE 3. Analysis of captured and transmitted images.

	Captured				Transmitted			
	JPEG image quality factor	image size	dimensions	metadata	image size	dimensions	metadata	Obfuscated code
Bank 1	95	700KB	1600×1200px	Exif	80KB	N.A.	N.A.	No
Bank 2	70	290KB	1600×1200px	Exif	290KB	1600×1200px	Exif	Yes
Bank 3	30	210KB	1600×1200px	Exif	80KB	1600×1125px	JFIF	Yes

captured by the camera while the third column represents the same information about the images transmitted to the servers; and finally, the fourth column shows the applications that use code obfuscation (discovered by inspection of the stack traces).

It can be noted from Table 3 that the App for Bank 1 compresses the image before sending it to the server, presumably to save bandwidth. In addition, its Exif metadata are replaced by JFIF metadata; the App for Bank 3 instead retrieves a low quality image from the camera from the start. We could not capture the transmitted images for Bank 3 using our Apache HTTP instrumentation. However, a total encrypted network traffic equal to approximately 80 *KB* per image was observed, suggesting that the images are sent over the network via some other mechanism. It was also discovered that OCR is performed by the mobile remote check deposit App on the smartphone as well.

The practice of sending low-quality images has important consequences for the server side's ability to detect forged images. In fact, while the regions of interest can still be processed successfully by OCR software, the loss of information in the high frequencies present in low quality JPEG images makes detection of artifacts introduced by forgeries hard to achieve. Indeed, pixel values tend to be more uniform across larger regions giving original images blocky features.

5) SERVER-SIDE OCR DECODING

Two experiments were designed to investigate server-side OCR decoding. The goal of first experiment was to determine the response of the server-side OCR decoding system in the presence of a mismatch between the user-entered amount and the automatically detected amount from the digital image of the check using OCR. To achieve this goal, we entered a different amount from the one written on the check that was entered during the user interaction step. It was observed from this experiment that the server-side OCR recognizes the mismatch and prompts the user to enter the correct amount again. If a different amount is entered again, the server temporarily accepts the amount entered by the user. Ultimately, however, the amount written on the check is deposited in the user's account. Findings of this experiment suggest that no OCR is being performed on the client side and that in the case of a disagreement between the amount entered by the user and the amount recognized by OCR, there is human intervention in the *check clearing system*.

The goal of second experiment was to determine the response of the server-side OCR decoding system in the presence of a mismatch between the user entered payee name and automatically detected payee name from the digital image of the check using OCR. To achieve this goal, intentionally misspelled payee names were written in the handwritten name portion of the check. It was observed through this experiment that the transaction proceeded without any glitches, suggesting that OCR is not performed on the handwritten name portion of the check.

The results of these two experiments suggest that the *check-clearing system* is configured to be tolerant towards errors, to the advantage of attackers.

Moreover, the wide variety of checks, lighting conditions in which pictures may be taken, and cameras justifies why it is difficult to set strict parameters for picture quality and JPEG characteristics on the server-side for check clearing system.

A. PERFORMANCE EVALUATION OF THE DIGITAL CHECK FORGERY ATTACK

To determine the effectiveness of the proposed digital check forgery attack, three checks with small amounts were tampered and injected into each selected application. During tampering characters of the original check were reused as outlined in Table 4 and described in § IV. In these experiments, the *payer* and the *payee* were the same person and the accounts were different accounts of that person with different banks or within the same bank. The tampered checks injected into the three applications were cleared without glitches within a business day.

By connecting the phone to a computer before a check transaction and switching among the different modes of operation described in the implementation section, the attack proceeds as follows.

- **Acquisition**: The camera subsystem is set in the *image saving* mode. The banking application for mobile remote check deposit is launched and a picture of the check front is acquired. The byte array with the image data and metadata is saved as a file on the local file system, along with being forwarded to the application. At this point, the transaction is canceled to avoid sending the original image to the server.

- **Digital forgery**: The saved image is copied from the smartphone using `adb` and, the procedure described in Section IV-A: **Library Instrumentation**. The copied image is then tampered in the forged check generation library in the Matlab environment. The tampered check is then pushed back onto the local file system of the smartphone.
- **Injection**: The camera subsystem is then placed in *image injection* mode.The banking application for mobile remote check deposit is launched and a picture of the check front is acquired again. The tampered image is loaded from the local file system as a byte array, which is forwarded to the application, while the byte array corresponding to the real image is blocked. Next, the camera subsystem is placed in *regular* mode and the picture of the check back is taken.
- **Forged check submission**: Finally, the images are submitted to the server.

This attack takes approximately ten minutes, most of which are spent by the user in pushing and pulling the image from the smart-phone and in providing input specifications to the Digital Forgery Library developed in the Matlab environment.

In our experiments, the percentage of the pixels altered was on average equal to 0.43% of the total number of pixels.

It was noted that due to the sensitive nature of the evaluation, the nature of various experiments we conducted were ''lightweight'' and our results have to be read in that light. More experiments and different forgeries are technically possible (forging account numbers, creating checks from scratch, using larger amounts) but have not been tested against any possible mitigation strategies currently employed by the banks due to their sensitive nature. More such experiments need to be done in collaboration with the banks to study the feasibility of these advanced attacks.

After successfully attacking the mRCD systems of the selected banks, we contacted the targeted banks, shared our findings and provided them with an earlier draft of this paper, nearly 5 months before submission of the conference version of this paper. The banks have acknowledged the problem of digital forgery and are actively working to design countermeasures. We also shared our preliminary ideas regarding countermeasures, which we discuss below.

VI. COUNTERMEASURES TO DIGITAL CHECK FORGERIES

Digital image forensic analysis methods can also be used for integrity verification of the digital image of the deposited check. Several techniques have been proposed for detecting forgeries in digital images [19], [20], [43]–[53]. The existing state of the art on image forensics [19], [20] relies on detecting inconsistencies in the statistical and geometric features of the input images to determine their authenticity.

Existing digital image forensic analysis techniques [19], [20] can be used to detect a wide range of forgeries including image cloning [43]–[45], image splicing [46]–[48], resampling [49]–[51], copy-move attack [52], [53] and so on. These techniques rely on resampling artifacts [49]–[51], color filter array (CFA) irregularities, lens aberrations and light inconsistencies [54]–[60] for forgery detection. A brief overview of existing image forensics techniques and their limitations in terms of the proposed attack model are discussed in the following.

- **Digital camera fingerprinting:** It has been shown in recent works [19], [20], [61]–[66], [68]–[71] that digital imaging sensors (e.g., CMOS sensors) leave artifacts (also known as photo response non-uniformity noise (PRUN) or camera noise) in the acquired pixel values. It has been demonstrated that camera noise is unique [19], [20], [61]–[71]; therefore, it can be used for camera fingerprinting. The PRUN-based fingerprint of an imaging device can be used for integrity verification of the digital image of the deposited check. To achieve this goal, the check-clearing server needs to extract the device imaging sensor fingerprint and compare it with the reference fingerprint. Estimating the reference fingerprint for each smartphone/check deposit system is a challenging task, which can be achieved by forcing each remote check deposit system to send a set of uncompressed images to the check-clearing server, for instance, by taking each a set of pictures and sending them when a check truncation application is first started. The check-clearing server can use these images for reference fingerprint estimation.
  One of the limitations of this countermeasure is that it can be defeated if an attacker transmit an initial set of images carrying the same fingerprint as the forged images to be used for a future attack. The effectiveness of these methods is therefore limited, as the reference camera noise fingerprint needed for integrity verification can be easily bypassed by using a ''one-time'' image acquisition device.
- **Color-filter array (CFA) estimation:** Modern digital cameras employ only a single color sensor at each pixel location, that is, for each pixel location only one color (red, green or blue) is measured directly instead of all three colors. A color filter array (CFA) is then used to interpolate the missing color samples to achieve a pixel-level three-channel color image. As a result, only one third of the samples in a color image are measured directly by the camera sensors and the remaining two thirds are estimated through using an interpolation process. The CFA interpolation process introduces specific correlations which are likely to be destroyed in the tampered image that can be used for forgery detection [19], [20], [44], [72]–[74].
  To test the effectiveness of these methods on forged check images, we applied three existing CFA-based image forgery detection methods [72]–[74] on the check images that were sent to the servers and noted that these methods failed to detect any forgery in those images. In fact, these forged checks were obtained through tampering followed by lossy compression; therefore,

TABLE 4. **Attack results.**

	Preliminary Experiment	Forgery Type	Outcome
Bank 1	Wrong Amount/names	Block Swapping Attack(Amount and Name Fields)	Successful
Bank 2	Wrong Amount	Block Swapping Attack(Amount Fields)	Successful
Bank 3	Wrong Amount	Block Swapping Attack(Amount Fields)	Successful

lossy compression artifacts suppress CFA interpolation artifacts contributing to poor performance for these methods.

- **Copy-evident images:** Copy-paste attacks introduce characteristic artifacts and distortions in the resulting JPEG files that are invisible to the naked eye but become visible when test images are recompressed under certain conditions [18]–[20], [75]. In regard to the check forgery detection problem, the signature artifacts need to be inserted before an attacker extracts the image. This goal can be achieved through camera hardware. Moreover, JPEG quantization tables verifiable on the server side can also be used to limit an attacker's ability to manipulate digital images. The proposed server-approved quantization table-based countermeasure has the potential to significantly reduce an attacker's chances of success.

  Check-forgery Detecting using Existing State of the Art: This experiment aims to evaluate performance of existing image forgery detecting detection approaches for check forgery detection. To this end, we have only considered the copy-paste attack vector. The motivation behind considering copy-paste attack is that it is the most common attack and can easily be applied over digital check-images. Existing copy-paste forgery detection (CPFD) methods [52], [53], [76] can be used to detect such forgeries. In order to validate the performance of the CPFD approaches for digital checks, we have copied different image regions in the original digital checks to modify the legal and courtesy amounts. A set of tampered check images are then analyzed using the DCT-based CPFD approach as proposed in [76]. CPFD method by Huang et. al. [76] computes texture features through DCT transform to capture traces of copy-paste attack. The homogeneous regions in terms of feature distance are then identified as duplicated regions. The DCT-based methods are the simplest texture representation methods and has their own limitations in terms of dealing with post processing operations over the forged regions, i.e., rotation, and flipping. However, as the region rotation, and flipping is not a common choice in check forgery attacks, therefore, these approaches are equally effective for check forgery attack detection. Shown in Figure 8, are the forged versions of the digital checks, and when we applied the DCT-based CPFD approach over the forged checks, it successfully identified the duplicated regions by computing the region similarity through the euclidean distance, and lexicographic sorting of the check regions [76]. One of the limitation of these approaches is that these approaches become ineffective when tampered regions are very small as compared to the total image size.
- **JPEG re-compression:** Many research efforts have been focused on characterizing, modeling, and extracting distortions due to double JPEG compression [14]–[17], [19], [20], [75], [77]. The digital image forensics community has proposed several techniques to capture traces of double JPEG compression [14]–[17], [19], [20], [75], [77]. It is important to highlight that existence of double compression artifacts alone is an insufficient condition to claim the presence of forgery. This is due to the fact that through our experiments it has been noted that existing mRCD apps do recompress check images before transmitting them to the check-clearing server. Furthermore, recent research efforts in the area of anti-forensics indicates that some of the double JPEG compression detection techniques can be defeated [78]. To address the limitations of existing image forgery detection approaches, an expert system based detection system is proposed here(see Fig. 9).

In evaluating these techniques against the image forgery methods devised in the Section III, it was observed that these methods are forgery specific, that is, a single method cannot be used to detect all possible check forgeries discussed in § III-D. For example, the camera fingerprinting method is not applicable if forged checks are deposited using different devices; similarly, demosaicing or CFA estimation-based methods are ineffective when images undergo a lossy compression, since the compression artifacts suppress the CFA interpolation artifacts.

To leverage the strength of different techniques, we propose an expert system-based forgery detection system consisting of N independent agents, where N is the number of forgery types that the proposed system can detect and $agent_i$ is designed to detect forgery type i. The decisions from all agents are fused to determine whether a check image is

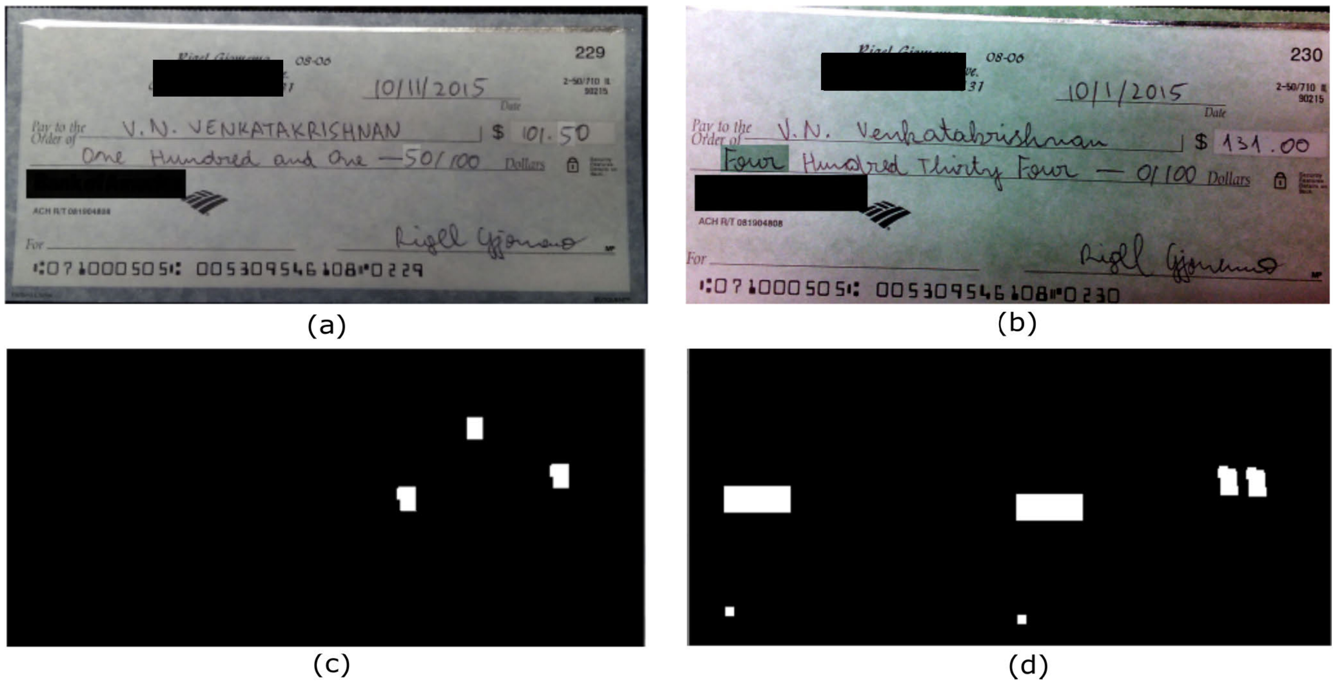


FIGURE 8. Copy move forgery detection. (a) and (b) are forged check images, whereas (c) and (d) are corresponding detected forged regions.

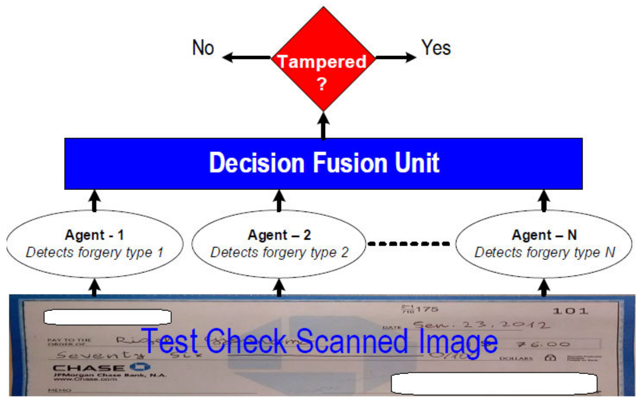


FIGURE 9. Shown is the block diagram of the proposed expert system-based forgery detector.

authentic or not. A conceptual block diagram of this proposed expert system-based framework is shown in Figure 9.

A. EXPERT SYSTEM-BASED DETECTOR FOR DIGITALLY FORGED CHECKS

This section provides an overview of the implementation details of the proposed expert system-based forgery detection system and its performance evaluation on real-world forged check images.

1) MULTI-AGENT-BASED DETECTOR: *IMPLEMENTATION*

The proposed expert system-based detector consisting of three agents is depicted in Figure 10. In particular, Agent-1 uses camera fingerprinting, Agent-2 uses local noise variance in the estimated CFA, and Agent-3 uses double JPEG compression artifacts for forgery detection. The decisions from these agents are then considered and weighed together in an *OR*-logic decision fusion unit.

Next, we provide the implementation details of each agent and the evaluation of our system in detecting forged checks.

- **Agent-1: Camera Fingerprint-based Forgery Detector**: Imaging sensors leave their characteristic fingerprint, also known as photo-response non-uniformity (PRNU) noise, in the images/video taken through them. The sensor PRNU noise can be used for various forensic tasks, including linking an image to a specific camera, integrity verification, copy-paste detection, and so on. For implementation and performance evaluation, the reference PRNU noise for each camera is estimated by averaging the noise obtained from multiple images using a denoising filter. Next, the reference PRNU noise and the noise residual extracted from the test image is used to link the image to a specific camera using a correlation detector. For forgery detection, Agent-1 uses local correlation between the reference PRNU noise and noise residual extracted from the test image. To realize Agent-1, we use an existing Matlab implementation of this method [61]–[69], [71].[3] This implementation is used with default settings on all three forged and corresponding original checks, whereas the reference PRNU noise template is estimated from the original check.
- **Agent-2: Modified CFA-based Forgery Detector**: This detection method uses disturbance in the

[3]http://dde.binghamton.edu/download/camera_fingerprint/

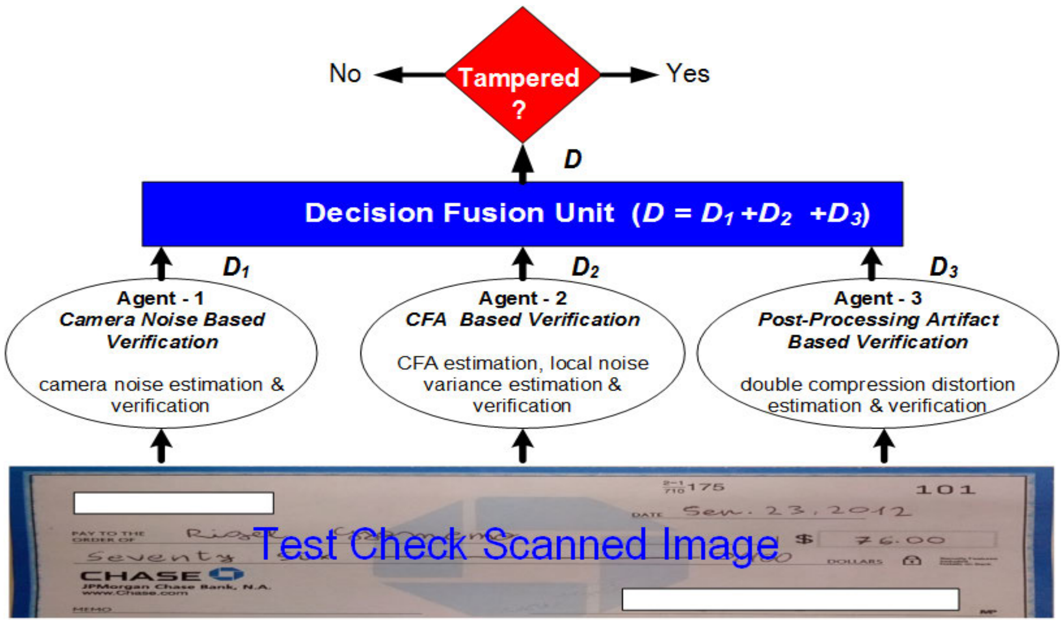

No
Tampered ?
Yes
D
Decision Fusion Unit (D = D1 +D2 +D3)
D1
D2
D3
Agent - 1
Camera Noise Based Verification
camera noise estimation & verification
Agent - 2
CFA Based Verification
CFA estimation, local noise variance estimation & verification
Agent - 3
Post-Processing Artifact Based Verification
double compression distortion estimation & verification
Test Check Scanned Image

FIGURE 10. Block diagram of a three-agent-based forgery detector.

correlations due to CFA interpolation. To achieve this goal, the proposed method first estimates the CFA from the test image. This estimated CFA, is next used to estimate the local noise variance for each color channel. Using this method, we observe that the forged writings on the check image start disappearing relatively sooner than the tampered ones. More specifically, threshold filtering is applied on the estimated noise variance for forgery detection.

The proposed modified CFA-based forgery detection method consists of three processing units: (i) CFA estimator, (ii) local noise variance estimator, and (iii) threshold filtering-based forgery detector. The conceptual block diagram of the proposed modified CFA-based method is shown in Figure 11.

1) **CFA Estimator**: Various methods have been proposed to estimate CFA [44], [72], [73]. In our implementation we use the cycle-spinning (CS)-based CFA method proposed in [74]. This method was selected due to its better performance compared with the other methods. In particular, in our system, we employ a Matlab implementation of the CS-based CFA estimation (downloaded from [74]). Next, we use each color channel of the estimated CFA, $f^c(i, j)$, $c \in \{r, g, b\}$ for the local noise variance estimator.
2) **Local noise variance estimator:** Image tampering disturbs the local noise structure. In particular, the variations in local image noise levels can become telltale evidence that the image has been tampered. To capture traces of local-noise variations, the near-constancy property of the kurtosis of an image in the band-pass filtered domain is used [79]. In particular, the estimated CFA $f^{(c)}(i, j)$, for each color channel, is transformed into the DCT domain. Next, the local variance σ and the kurtosis κ for each selected coefficient (i, j) are computed using rectangular windows surroundings (i, j), $\Omega_{(i,j)}$ as in [48],
$$\sigma^2_{(i,j)} = \mu_2 - \mu_1^2 \tag{3}$$
and
$$\kappa_{(i,j)} = \frac{\mu_4 - 4\mu_3\mu_1 + 6\mu_2\mu_1^2 - 3\mu_1^4}{\mu_2^2 - 2\mu_2\mu_1^2 + \mu_1^4} - 3 \tag{4}$$
where μ_m denotes m^{th} un-centered moment defined as,
$$\mu_m(\Omega_{(i,j)}) = \frac{1}{|\,\Omega_{(i,j)^k}|} \sum_{(\acute{i},\acute{j})\in\Omega_{(i,j)}} f(\acute{i}, \acute{j}, k)^m \tag{5}$$
Here, $f(\acute{i}, \acute{j}, k)$ is the response at $(\acute{i}, \acute{j})$ in the k^{th} subband of the transformed mask.
3) **Threshold filtering-based forgery detection:** Tampered regions are detected using threshold filtering of the estimated local noise variances. To this end, an adaptive set of threshold values obtained from estimated local noise variances is used for threshold filtering. In our tests, we observe that during threshold filtering, handwriting disappear uniformly for original checks, whereas, for a forged checks handwriting disappears non-uniformly. More specifically, during threshold filtering, forged handwriting start disappearing earlier than the untampered handwriting (in the same check).

- **Agent-3: JPEG Artifacts-based Forgery Detector**: To capture traces of double JPEG compression, Agent-3

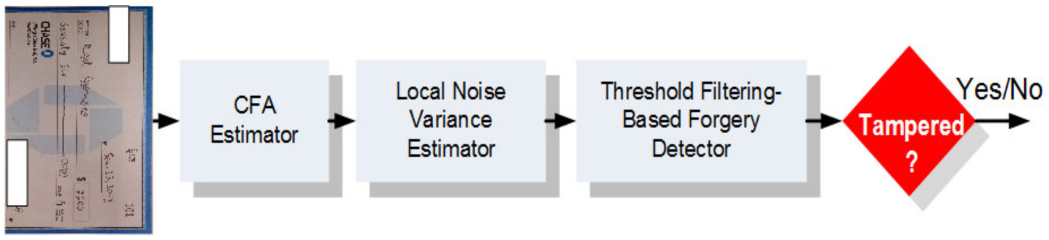

CFA Estimator
Local Noise Variance Estimator
Threshold Filtering-Based Forgery Detector
Tampered ?
Yes/No

FIGURE 11. Shown is the block diagram of the proposed modified CFA-based forgery detector.

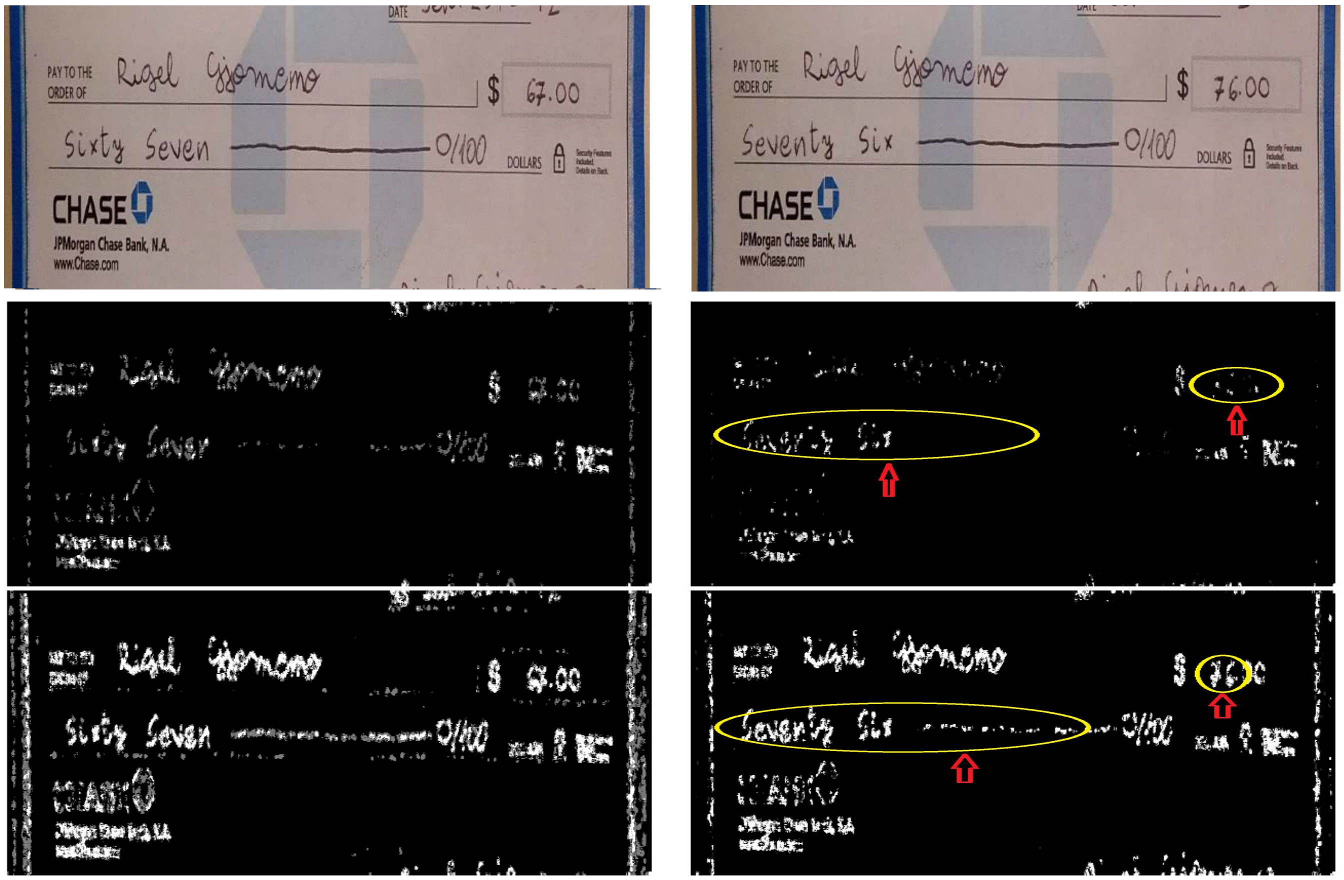


FIGURE 12. Shown in the first row are the original (left col.) and tampered (right col.) check images used to attack Bank1; and shown in the second and third rows are the filtered estimated local-noise from these images using the proposed method with threshold values {.26 *and* .3}; the estimated noise profile in tampered regions is highlighted using yellow circles and red arrows.

relies on the blind forgery localization in JPEG images proposed in [75]. This method does not require any prior information about the location of the manipulated area. The method uses a unified statistical model to characterize DCT coefficients when double JPEG compression is applied which is used for the generation of a likelihood map that shows the probability of each 8×8 image block being doubly compressed.

Agent-3 was realized using a Matlab implementation of this method [4] and used with default settings to analyze all three forged and corresponding original checks.

2) PERFORMANCE EVALUATION: AGENT-LEVEL

To evaluate our system, we have conducted two types of experiments: (1) using the images that were actually submitted to the banks, and (2) using a larger dataset of forged images, which were not submitted to the banks due to the sensitive nature of the evaluation. We present them in order next.

- **Agent-1**: During our experiments, Agent-1 was able to detect the inconsistencies in all three forged checks. However, this method has limited applicability since the reference PRNU for each device must be made available to the banks. More specifically, the reference PRNU is estimated from 10 images from the same cell-phone camera using Matlab code downloaded from[5] with default settings. For tamper detection, the PRNU is estimated from the input check image using the default settings of Goljan *et al.*'s algorithm [61]–[66], [71]. Block-level correlation is then computed between

[4]http://lesc.det.unifi.it/en/node/187

[5]http://dde.binghamton.edu/download/camera_fingerprint/

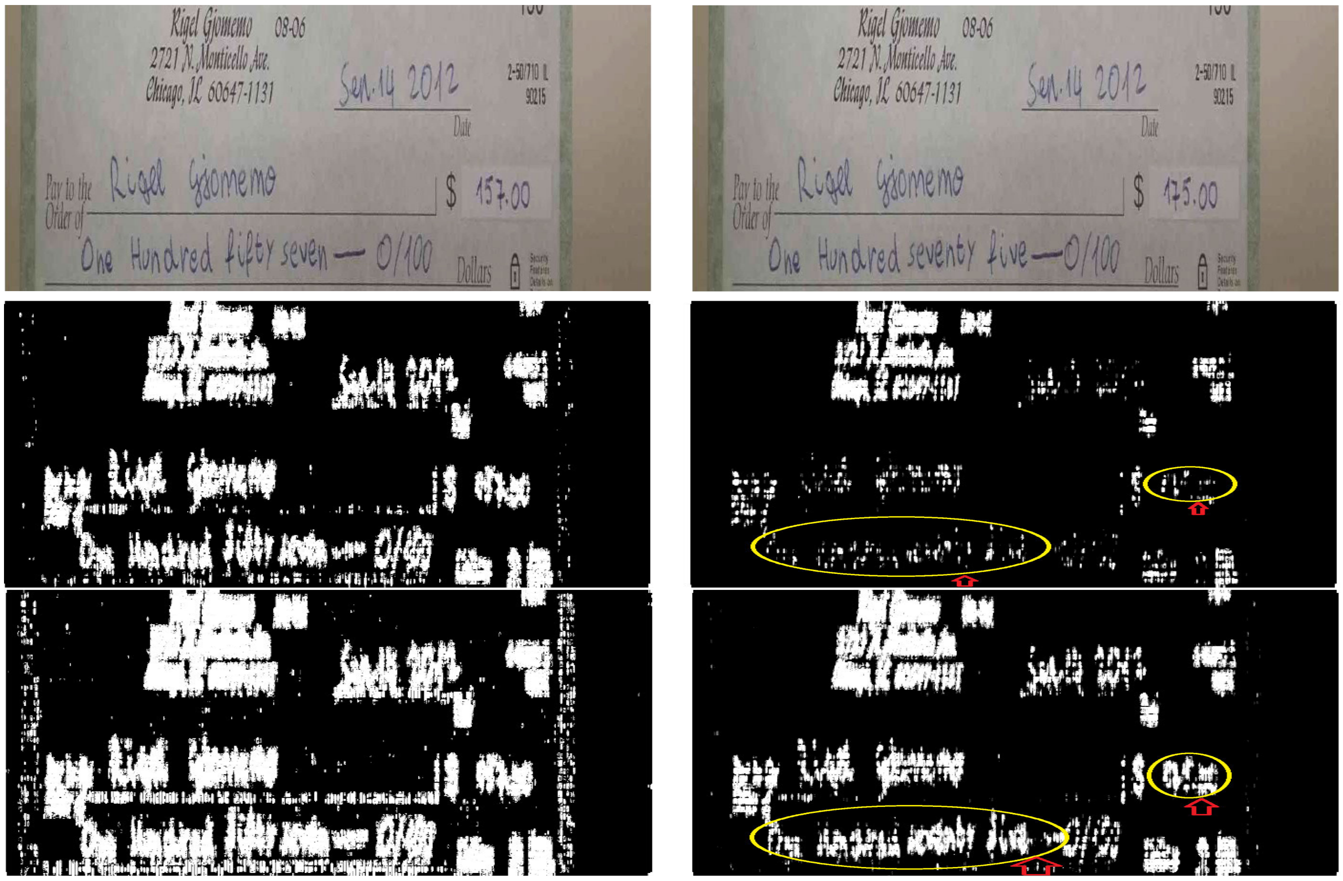


FIGURE 13. Shown in the first row are the original (left col.) and tampered (right col.) check images used to attack Bank2; and shown in the second and third rows are the filtered estimated local noise from these images using the proposed method with threshold values {.26 *and* .3}; estimated noise profile in the tampered regions is highlighted using yellow circles and red arrows.

the reference PRNU and the estimated PRNU from the test image, which is used for forgery detection. It has been observed that for forged blocks the normalized correlation value was less than 0.5, and for original blocks this value was grater than 0.8. With the threshold value of 0.6, Agent-1 was able to detect all forged checks.

- **Agent-2**: To evaluate the thresholded approach of Agent-2, binary images thresholded for $\tau = \{0.24, .25, \cdots, 0.3\} \times \min(L_{nv})$ are shown in Figures 12– 14. It can be observed from Figures 12 - 14 that for all three banks, the local variance around handwriting, especially, in the legal and courtesy areas is different from the rest of the check. The variation in the estimated local noise variances around handwritings is used for forgery detection.
- **Agent-3**: The implementation of Agent-3 was able to localize the forgery in the forged check deposited to Bank-3, whereas for the remaining two forged checks it was unable to localize the forgery. Low-quality JPEG image acquisition for the Bank-3 App for remote check deposit seems to be the reason behind these split results. Shown in Figure 15 are the detection results for all three forged checks analyzed using the double JPEG compression detection method discussed in [75]. It can be observed from Figure 15 that the selected method is able to localize the tampering location for Bank-3, whereas for the other two banks it is unable to localize the tampering location.

It is clear from the preliminary results that existing solutions are limited in their scope and application. It can also be observed from Figures 12– 15 that the proposed expert system-based forgery detector is capable of detecting check forgeries in checks injected to all three banks. Flexibility to incorporate future solutions for digital image tamper detection is one of salient features of the proposed framework.

B. PERFORMANCE EVALUATION: SYSTEM-LEVEL

The goal of this experiment is to evaluate the performance of our system on a relatively larger dataset of forged checks. We note that we did not submit these forged checks to the banks due to the sensitive nature of the experiments. As there is no dataset containing images of actual checks in the public domain, we decided to generate one. To this end, we collected images of three different checks using three different smartphones under two lighting conditions. These 15 original images of checks were then used to generate forged check

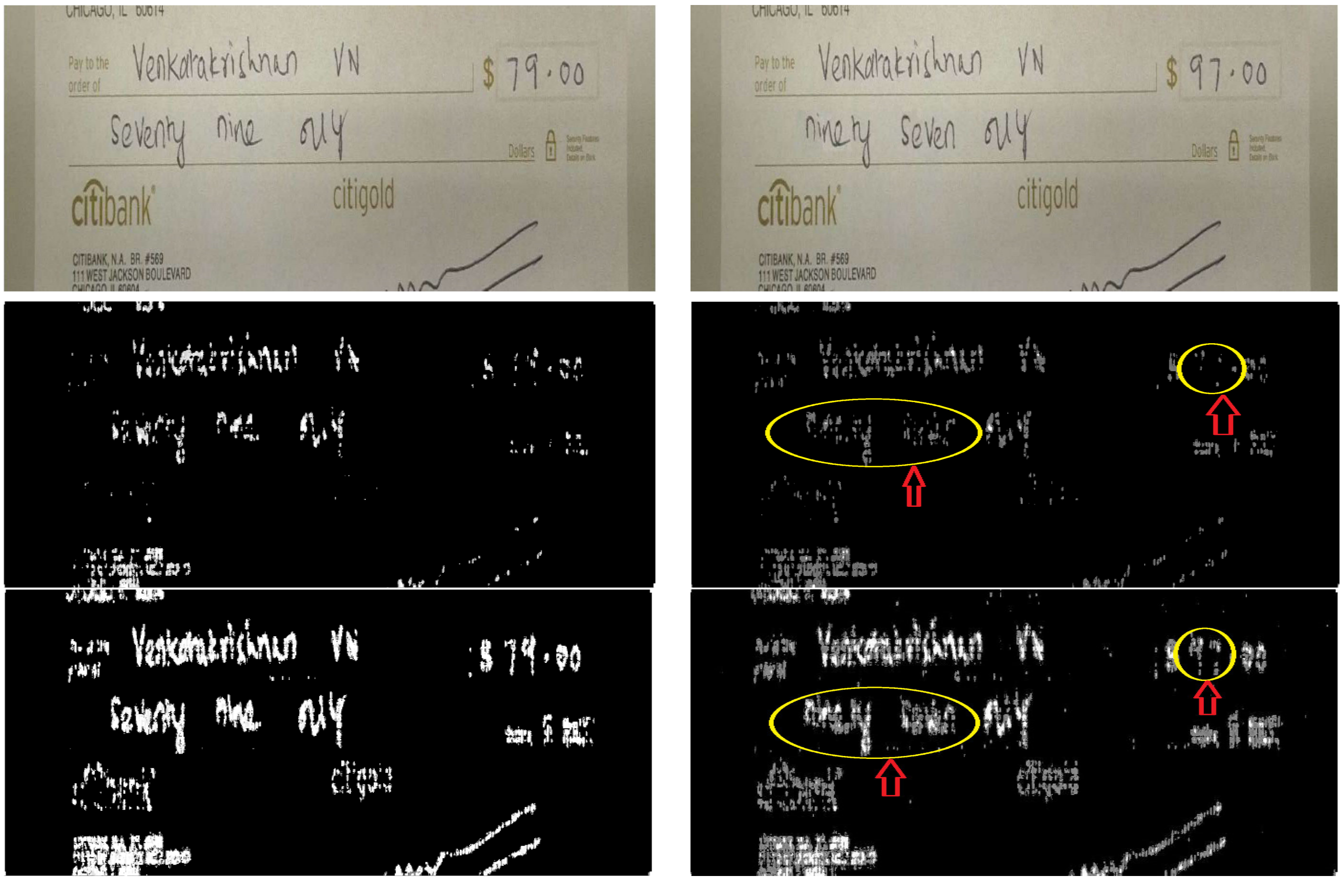


FIGURE 14. Shown in the first row are the original (left col.) and tampered (right col.) check images used to attack Bank3; and shown in the second and third rows are the filtered estimated local noise from these images using the proposed method with threshold values {.26 *and* .3}; the estimated noise profile in the tampered regions is highlighted using yellow circles and red arrows.

images. For forged image generation we considered the following check fields:

1) Date field
2) Payee name field
3) Amount (in numbers) also known as the courtesy amount field
4) Written amount also known as the legal amount field
5) Account number field
6) Check number field
7) Routing number field
8) Signature field

A forged check image is obtained by tampering one or more of selected check field(s). More specifically, a forged check is obtained by replacing selected check field(s) with text(s) from the same check image or from other check image(s). To this end, a set of 100 forged check images was generated from 15 original check images with varying levels of tampering ratios (TR) - *a ratio of the number of modified pixels to the total number of pixels*, e.g., $0.25\% \leq TR < 1\%$. The set of forged check images was used for performance evaluation. The proposed expert system-based forgery detector (shown in Fig. 6) was tested on the forged check image dataset, and it achieved overall detection accuracy of 100%.

Moreover, performance of each forgery detection agent separately was also evaluated. Agent-level detection performance analysis indicated that Agent-1 (camera fingerprinting based detection) achieved detection accuracy of 98%, Agent-2 (local noise variance in the estimated CFA based detection) achieved perfect detection accuracy, i.e, 100%, and Agent-3 (double JPEG compression artifacts based detection) achieved detection accuracy of 81%. We observe that the detection accuracy of Agent-3 depends on the tampering ratio. It has been observed through experimentation that Agent-3 failed to detect forgeries whenever the tampered check image contains $TR < 1\%$.

It have been observed through performance evaluation of the selected dataset that the proposed expert system-based forgery detector is effective in detecting digital forgeries in the check images with very high accuracy.

C. ADDITIONAL COUNTERMEASURES

In addition to the digital image forensics techniques described earlier, other countermeasures may be used to combat digital check forgeries. These countermeasures can be deployed either on the client side or on the server side. This section provides a brief overview of additional countermeasures that can be used to detect digital check forgery attacks.

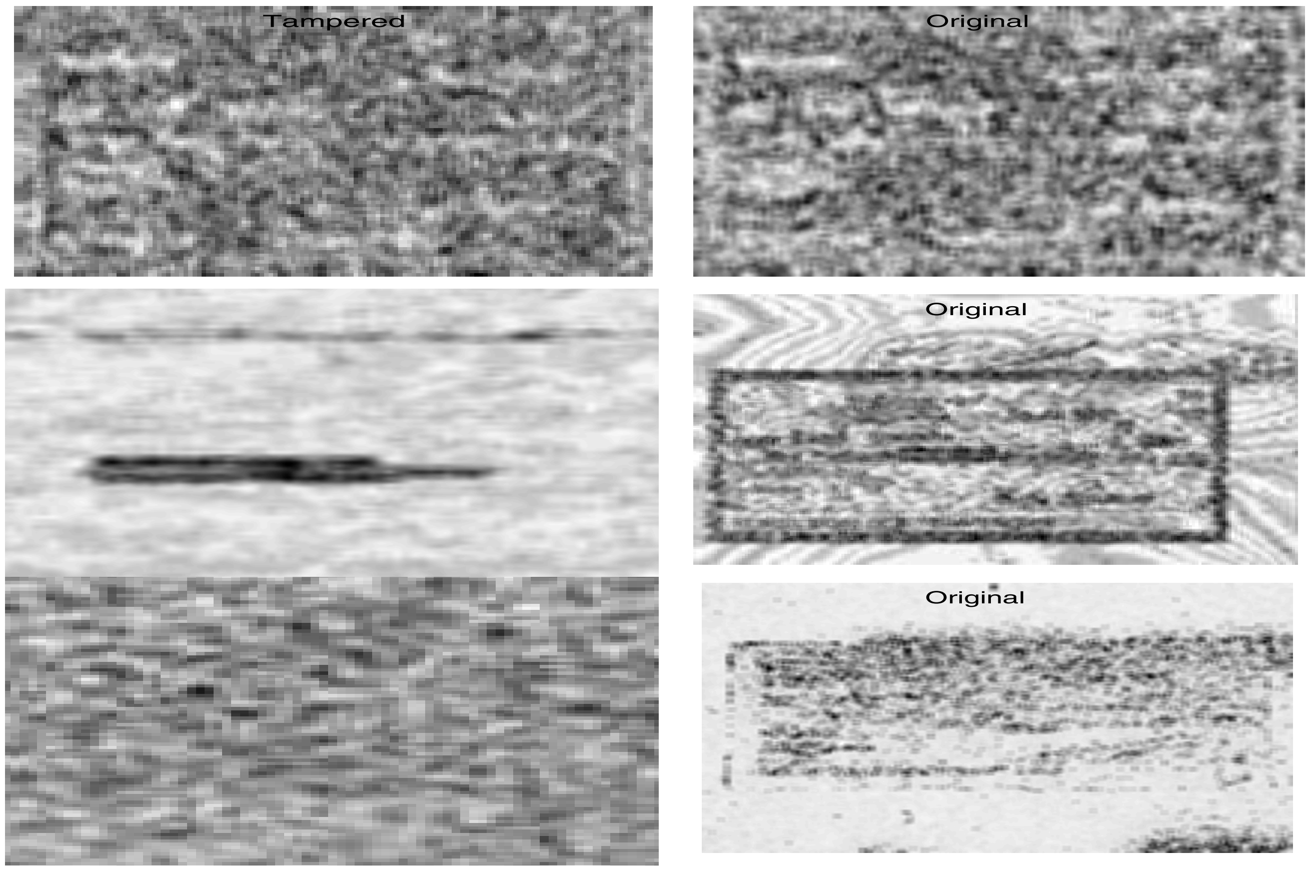


FIGURE 15. Shown in the left column (from top to bottom) are the double compression artifacts estimated from tampered check images injected into Bank-1 to Bank-3, and shown in right column (from top to bottom) are the double compression artifacts estimated from the corresponding original images.

1) TRUSTED COMPUTING (TC)

Trusted computing (TC) (also known as trustworthy computing) aims to resolve computer security issues by enhancing hardware and associated software modifications. With TC, the computing device is predicted to consistently behave in expected ways, and predicted behaviors are enforced through hardware enhancements and software modifications. Predicted behavior of the computing device is achieved by loading enhanced hardware with a unique encryption key inaccessible to the rest of the system. The term trusted computing generally refers to the industry ideal of a computing system with built-in security mechanisms that place minimal reliance on the end user to keep the machine and its peripheral devices secure.

The TC-based solution is not a new concept; it has been proposed and implemented to prevent client-side tampering in a wide range of devices [80]–[82]. A trusted computing platform, therefore, can be used to prevent client-side digital image tampering. The proposed TC platform can be used to digitally sign either the entire acquired check-image data or the sensitive portions of the check-image (i.e., legal and courtesy amounts, account number) using a hardware-based tamper-resistant and trusted module on that platform before releasing the image to the upper layers of the OS.

To realize this goal, the OMAP 4 hardware platform on the Galaxy Nexus smartphone (used for the experimental results) and OMAP 5 hardware platform on latest Samsung Galaxy smartphone models provide capabilities to implement a trusted computing module on these smartphones [83]. To fully utilize the available capabilities and resources on the OMAP X hardware platform, the mRCD applications must be modified to interface with the trusted module. It is important to highlight that these solutions are still vulnerable to attacks that alter the physical check before digital image acquisition.

2) CHECK RECONCILIATION-BASED APPROACHES

Check reconciliation is one of the oldest and most effective techniques for consumers to prevent check fraud. It relies on keeping a record of the check issued which is then used to match against the actual transaction record. It has been noted that a large fraction of users do not reconcile their accounts [84].

3) POSITIVE PAY

Check reconciliation is the main motivation behind *positive pay*—a countermeasure commonly used to protect against check forgery developed primarily for businesses. Positive pay is a well-established and effective protective counter measure against check forgery. In positive pay, the payee provides reference copy(ies) of the original check(s) by sending copies of the issued checks to the bank everyday. The cost associated with positive pay is one of the factors behind this countermeasure being used only by corporations and companies. For instance, according to an Association for Financial Professionals survey [9]. 29% of the surveyed companies do not use it due to the associated costs.

4) TRANSACTION LIMITS

Transaction limit is another proactive countermeasure that aims to reduce the risk associated with remote check fraud. Currently, financial institutions introduce a daily as well as a monthly amount limits on remote check deposit transactions. This countermeasure does minimize the impact of the financial loss due to digital check forgery, but unfortunately, it does not prevent proposed attacks. Moreover, this countermeasure seems to be available only to individual customers but is currently unavailable to businesses.

5) DYNAMIC WATERMARKING

To further improve the chances of forgery detection on the server side, the mRCD App can be modified to acquire raw(uncompressed) images of the regions of interest (ROIs) (e.g., courtesy amount, legal amount, bank routing number, account number, etc.), embed dynamic watermarks in them using robust watermarking, and transmit them to the server side. This is a simple safeguard to implement, but its effectiveness can be realized only by developing and deploying the appropriate forgery detection framework on the server side.

VII. CONCLUSION

In this paper, we have highlighted vulnerabilities in the existing remote check deposit system and presented an attack model based on a digital check forgery attack. The proposed attack vectors can be used to exploit vulnerabilities in the existing client check truncation systems. The proposed attack vectors are enabled by the delegation to untrusted entities of critical operations performed by trusted entities. In this paper, the feasibility of an instance of this attack has been successfully demonstrated with experiments on three banking applications of leading banks in the US running on Android smartphones. We have also outlined countermeasures to safeguard check-clearing system against the proposed attack vectors. Specifically, we have provided an expert system-based forgery detector. Performance evaluation for the proposed expert system-based forgery detector against digital check forgery attack is also provided. A brief list of additional countermeasures to safeguard remote check-deposit systems against tampered check image injection attacks is also proposed.

Our ongoing research efforts are focused on investigating the impact of machine learning and artificial intelligence on security of the remote check-deposit systems and the use of advanced machine-learning techniques, e.g., deep learning for forgery detection.

REFERENCES

[1] B. Meara. (2015). *State of Remote Deposit Capture 2015: Mobile is the New Scanner*. [Online]. Available: https://www.celent.com/insights/572842967

[2] P. Murphy. *Looking Back on 2016*. Accessed: Dec. 2016. [Online]. Available: https://www.remotedepositcapture.com/news/Looking-Back-On-2016.aspx

[3] *Check Clearing for the 21st Century Act (Check 21)*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.fdic.gov/consumers/consumer/alerts/check21.html

[4] *Merchant Remote Deposit Capture*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.marketsandmarkets.com/PressReleases/merchant-remote-deposit-capture.asp

[5] *How Popular is Mobile Check Deposit?* Accessed: Mar. 25, 2020. [Online]. Available: http://www.pymnts.com/briefing-room/mobile/playmakers/2013/how-popular-is-mobile-check-deposit/

[6] *The State of Mobile Banking 2012*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.forrester.com/The+State+Of+Mobile+Banking+2012/fulltext/-/E-RES75582

[7] *Banks to Accept Smartphone Photos of Cheques*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.theguardian.com/money/2013/dec/26/banks-smartphone-photos-cheques

[8] *Canadian Banks Get Ok to Move Ahead With Image Cheque Deposit*. Accessed: Mar. 2020. [Online]. Available: http://www.canadianlawyermag.com/legalfeeds/1640/canadian-banks-get-ok-to-move-ahead-with-image-cheque-deposit.html

[9] *Check Fraud Survey*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.afponline.org/fraud/

[10] *Account Deposit Survey*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.aba.com/Products/Surveys/Pages/2013DepositAccount.aspx

[11] R. Gjomemo, H. Malik, N. Sumb, V. N. Venkatakrishnan, and R. Ansari, "Digital check forgery attacks on client check truncation systems," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Berlin, Germany: Springer, 2014, pp. 3–20.

[12] [Online]. Available: http://www.ffiec.gov/exam/check21/faq.htm#General

[13] H. Yu, T.-T. Ng, and Q. Sun, "Recaptured photo detection using specularity distribution," in *Proc. 15th IEEE Int. Conf. Image Process.*, Oct. 2008, pp. 3140–3143.

[14] H. Farid, "Exposing digital forgeries from JPEG ghosts," *IEEE Trans. Inf. Forensics Security*, vol. 4, no. 1, pp. 154–160, Mar. 2009.

[15] T. Bianchi and A. Piva, "Detection of nonaligned double JPEG compression based on integer periodicity maps," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 842–848, Apr. 2012.

[16] T. Pevny and J. Fridrich, "Detection of double-compression in JPEG images for applications in steganography," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 2, pp. 247–258, Jun. 2008.

[17] B. Li, Y. Q. Shi, and J. Huang, "Detecting doubly compressed JPEG images by using mode based first digit features," in *Proc. IEEE 10th Workshop Multimedia Signal Process.*, Oct. 2008, pp. 730–735.

[18] A. Lewis and M. Kuhn, "Towards copy-evident jpeg images," *GI Jahrestagung*, vol. 154, pp. 1582–1591, Jan. 2009.

[19] H. Farid, "Image forgery detection—A survey," *IEEE Signal Process. Mag.*, vo. 26, no. 2, pp. 16–25, Apr. 2009.

[20] A. Piva, "An overview on image forensics," *ISRN Signal Process.*, vol. 2013, pp. 1–22, Jan. 2013.

[21] L. Zhou, D. Wang, Y. Guo, and J. Zhang, "Blur detection of digital forgery using mathematical morphology," in *Proc. KES Int. Symp. Agent Multi-Agent Syst., Technol. Appl.* Berlin, Germany: Springer, 2007, pp. 990–998.

[22] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.

[23] G. K. Birajdar and V. H. Mankar, "Digital image forgery detection using passive techniques: A survey," *Digit. Invest.*, vol. 10, no. 3, pp. 226–245, Oct. 2013.

[24] R. Böhme and M. Kirchner, *Counter-Forensics: Attacking Image Forensics*. New York, NY, USA: Springer, 2013, pp. 327–366.

[25] M. A. Ferrer, M. Diaz, C. Carmona-Duarte, and A. Morales, "A behavioral handwriting model for static and dynamic signature synthesis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1041–1053, Jun. 2017.

[26] C. Rabasse, R. M. Guest, and M. C. Fairhurst, "A new method for the synthesis of signature data with natural variability," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 3, pp. 691–699, Jun. 2008.

[27] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Offline handwritten signature verification—Literature review," in *Proc. 7th Int. Conf. Image Process. Theory, Tools Appl. (IPTA)*, Nov. 2017, pp. 1–8.

[28] M. A. Ferrer, M. Diaz, C. Carmona-Duarte, and R. Plamondon, "A biometric attack case based on signature synthesis," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2018, pp. 1–6.

[29] M. A. Ferrer, M. Diaz, and C. Carmona-Duarte, "Two-steps perceptual important points estimator in 8-connected curves from handwritten signature," in *Proc. 7th Int. Conf. Image Process. Theory, Tools Appl. (IPTA)*, Nov. 2017, pp. 1–5.

[30] W. Li, L. Zhao, Z. Lin, D. Xu, and D. Lu, "Non-local image inpainting using low-rank matrix completion," *Comput. Graph. Forum*, vol. 34, no. 6, pp. 111–122, Sep. 2015.

[31] V. Kumar, J. Mukherjee, and S. K. D. Mandal, "Image inpainting through metric labeling via guided patch mixing," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5212–5226, Nov. 2016.

[32] T. Nakamura, A. Zhu, K. Yanai, and S. Uchida, "Scene text eraser," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Nov. 2017, pp. 832–837.

[33] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1520–1528.

[34] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004.

[35] R. Palacios and A. Gupta, "A system for processing handwritten bank checks automatically," *Image Vis. Comput.*, vol. 26, no. 10, pp. 1297–1313, Oct. 2008.

[36] N. Efford, *Digital Image Processing: A Practical Introduction Using Java*. London, U.K.: Pearson, 2000.

[37] A. Chowriappa, R. N. Rodrigues, T. Kesavadas, V. Govindaraju, and A. Bisantz, "Generation of handwriting by active shape modeling and global local approximation (GLA) adaptation," in *Proc. 12th Int. Conf. Frontiers Handwriting Recognit.*, Nov. 2010, pp. 206–211.

[38] S. C. Park, M. K. Park, and M. G. Kang, "Super-resolution image reconstruction: A technical overview," *IEEE Signal Process. Mag.*, vol. 20, no. 3, pp. 21–36, May 2003.

[39] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.

[40] G. Anbarjafari and H. Demirel, "Image super resolution based on interpolation of wavelet domain high frequency subbands and the spatial domain input image," *ETRI J.*, vol. 32, no. 3, pp. 390–394, Jun. 2010.

[41] *ExifTool By Phil Harvey*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.sno.phy.queensu.ca/~phil/exiftool/

[42] *Android Open Source Project*. Accessed: Mar. 25, 2020. [Online]. Available: http://source.android.com/

[43] M. K. Johnson and H. Farid, "Exposing digital forgeries in complex lighting environments," *IEEE Trans. Inf. Forensics Security*, vol. 2, no. 3, pp. 450–461, Sep. 2007.

[44] A. C. Popescu and H. Farid, "Exposing digital forgeries by detecting duplicated image regions," Dept. Comput. Sci., Dartmouth College, Hanover, NH, USA, Tech. Rep., Apr. 2004.

[45] H. Ge and H. Malik, "Exposing image forgery using inconsistent reflection vanishing point," in *Proc. IEEE/IET Int. Conf. Audio, Lang., Image Process. (ICALIP)*, Shanghai, China, Jul. 2014, pp. 282–286.

[46] T. Ng and S. Chang, "A model for image splicing," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, vol. 2, Oct. 2004, pp. 1257–1260.

[47] W. Wang, J. Dong, and T. Tan, "Effective image splicing detection based on image chroma," in *Proc. 16th IEEE Int. Conf. Image Process. (ICIP)*, Nov. 2009, pp. 1257–1260.

[48] X. Pan, X. Zhang, and S. Lyu, "Exposing image splicing with inconsistent local noise variances," in *Proc. IEEE Int. Conf. Comput. Photography (ICCP)*, Apr. 2012, pp. 1–10.

[49] A. C. Popescu and H. Farid, "Exposing digital forgeries by detecting traces of resampling," *IEEE Trans. Signal Process.*, vol. 53, no. 2, pp. 758–767, Feb. 2005.

[50] D. Vazquez-Padin and F. Perez-Gonzalez, "Prefilter design for forensic resampling estimation," in *Proc. IEEE Int. Workshop Inf. Forensics Secur.*, Nov. 2011, pp. 1–6.

[51] M. Kirchner and T. Gloe, "On resampling detection in re-compressed images," in *Proc. 1st IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2009, pp. 21–25.

[52] X. Pan and S. Lyu, "Region duplication detection using image feature matching," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 4, pp. 857–867, Dec. 2010.

[53] J. Fridrich, D. Soukal, and J. Lukas, "Detection of copy-move forgery in digital images," in *Proc. Digit. Forensic Res. Workshop*, 2003, pp. 1–10.

[54] J. F. O'brien and H. Farid, "Exposing photo manipulation with inconsistent reflections," *ACM Trans. Graph.*, vol. 31, no. 1, pp. 1–11, Jan. 2012.

[55] E. Kee, J. O'Brien, and H. Farid, "Exposing photo manipulation with inconsistent shadows," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 1–12, Jun. 2013.

[56] W. Zhang, X. Cao, J. Zhang, J. Zhu, and P. Wang, "Detecting photographic composites using shadows," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jun. 2009, pp. 1042–1045.

[57] C. Riess and E. Angelopoulou, "Scene illumination as an indicator of image manipulation," in *Proc. 12th Int. Workshop Inf. Hiding*, 2010, pp. 1042–1045.

[58] E. Kee and H. Farid, "Exposing digital forgeries from 3-D lighting environments," in *Proc. IEEE Int. Workshop Inf. Forensics Secur.*, Dec. 2010, pp. 1042–1045.

[59] M. K. Johnson and H. Farid, "Exposing digital forgeries through chromatic aberration," in *Proc. 8th workshop Multimedia Secur. (MM Sec)*, Geneva, Switzerland, 2006, pp. 48–55.

[60] T. Gloe, A. Winkler, and K. Borowka, "Efficient estimation and large-scale evaluation of lateral chromatic aberration for digital image forensics," in *Proc. SPIE 12th Electron. Imag., Steganography, Secur., Watermarking Multimedia Contents*, N. Memon, J. Dittmann, and A. Alattar, Eds., 2010, Art. no. 754107.

[61] M. Goljan, J. Fridrich, and T. Filler, "Large scale test of sensor fingerprint camera identification," in *Proc. SPIE, 11th Electron. Imag., Media Forensics Secur.*, vol. 7254, N. Memon, E. Delp, P. Wong, and J. Dittmann, Eds. Bellingham, WA, USA: SPIE, 2009, pp. 0I-1–0I-12.

[62] M. Goljan and J. Fridrich, "Camera identification from cropped and scaled images," in *Proc. SPIE, 10th Electron. Imag., Secur., Forensics, Steganography, Watermarking Multimedia Contents*, vol. 6819. Bellingham, WA, USA: SPIE, 2008, Art. no. 68190E.

[63] M. Goljan, J. Fridrich, and M. Chen, "Defending against fingerprint-copy attack in sensor-based camera identification," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 1, pp. 227–236, Mar. 2011.

[64] M. Goljan, J. Fridrich, and T. Filler, "Managing a large database of camera fingerprints," in *Proc. SPIE, 12th Electron. Imag., Steganography, Secur., Watermarking Multimedia Contents*, vol. 7541, N. Memon, J. Dittmann, and A. Alattar, Eds. Bellingham, WA, USA: SPIE, 2010, pp. 08-1–08-12.

[65] M. Goljan, J. Fridrich, and M. Chen, "Countering counter-forensics," in *Proc. SPIE, 12th Electron. Imag., Steganography, Secur., Watermarking Multimedia Contents*, vol. 7541, N. Memon, J. Dittmann, and A. Alattar, Eds. Bellingham, WA, USA: SPIE, 2010, pp. 0S-1–0S-12.

[66] J. Fridrich and M. Goljan, "Determining approximate age of digital images using sensor defects," in *Proc. 3rd Media Watermarking, Secur., Forensics*, Feb. 2011, Art. no. 788006.

[67] M. Chen, J. Fridrich, M. Goljan, and J. Lukas, "Determining image origin and integrity using sensor noise," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 1, pp. 74–90, Sep. 2008.

[68] Z. Lint, R. Wang, X. Tang, and H.-Y. Shum, "Detecting doctored images using camera response normality and consistency," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, San Diego, CA, USA, Jun. 2005, pp. 1087–1092.

[69] S. Bayram, H. T. Sencar, and N. Memon, "Scale invariance and noise in natural images," in *Proc. IEEE Western New York Image Process. Workshop*, Sep. 2009, pp. 2209–2216.

[70] J. Luka, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 2, pp. 205–214, Jun. 2006.

[71] M. Goljan and J. Fridrich, "Sensor-fingerprint based identification of images corrected for lens distortion," in *Proc. SPIE, Electron, Media Watermarking, Secur., Forensics*, Feb. 2012, Art. no. 83030H.

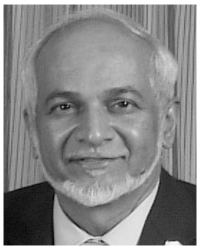
[72] A. C. Popescu and H. Farid, "Exposing digital forgeries in color filter array interpolated images," *IEEE Trans. Signal Process.*, vol. 53, no. 10, pp. 3948–3959, Oct. 2005.

[73] M. Kirchner, "Efficient estimation of CFA pattern configuration in digital camera images," in *Proc. SPIE, 12th Electron. Imag., Steganography, Secur., Watermarking Multimedia Contents*, N. Memon, J. Dittmann, and A. Alattar, Eds., 2010, Art. no. 754111.
[74] J. Tian, W. Yu, and L. Ma, "Color filter array color reproduction using cycle-spinning," *AEU-Int. J. Electron. Commun.*, vol. 64, no. 6, pp. 584–587, Jun. 2010. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/28174-color-filter-array-color-reproduction-using-cycle-spinning
[75] T. Bianchi and A. Piva, "Image forgery localization via block-grained analysis of JPEG artifacts," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 1003–1017, Jun. 2012.
[76] Y. Huang, W. Lu, W. Sun, and D. Long, "Improved DCT-based detection of copy-move forgery in images," *Forensic Sci. Int.*, vol. 206, nos. 1–3, pp. 178–184, Mar. 2011.
[77] C. Chen, Y. Q. Shi, and W. Su, "A machine learning based scheme for double JPEG compression detection," in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.
[78] M. C. Stamm and K. J. R. Liu, "Anti-forensics of digital image compression," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 1050–1065, Sep. 2011.
[79] D. Zoran and Y. Weiss, "Scale invariance and noise in natural images," in *Proc. IEEE Int. Conf. Comput. Vis.* Kyoto, Japan: IEEE, Sep./Oct. 2009, pp. 2209–2216.
[80] E. Gallery, *An Overview of Trusted Computing Technology*, vol. 6. Edison, NJ, USA: IET, 2005.
[81] *Trusted Computing Group*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.trustedcomputinggroup.org/
[82] *Jelly Bean Hardware-Backed Credential Storage*. Accessed: Mar. 25, 2020. [Online]. Available: http://nelenkov.blogspot.com/2012/07/jelly-bean-hardware-backed-credential.html
[83] *mShield*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.ti.com/pdfs/wtbu/ti_mshield_whitepaper.pdf
[84] *A Balancing Act*. Accessed: Mar. 25, 2020. [Online]. Available: http://www.moebs.com/AboutUs/Moebsinthenews/tabid/57/ctl/Details/mid/484/ItemID/26/Default.aspx

HAFIZ MALIK (Senior Member, IEEE) received the B.E. degree in electrical engineering from the University of Engineering and Technology Lahore, Lahore, Pakistan, in 1999, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Chicago, Chicago, IL, USA, in 2006. He has been with the University of Michigan–Dearborn, since 2007. He is currently an Associate Professor with the Department of Electrical and Computer Engineering (ECE), University of Michigan–Dearborn. He has published more than 100 articles in leading journals, conferences, and workshops. He is on the Review Board Committee of IEEE Technical Committee on Multimedia Communications (MMTC). He organized Special Track on Doctoral Dissertation in Multimedia, in the 6th IEEE International Symposium on Multimedia (ISM), in 2006. He is also organizing a special session on Data Mining in Industrial Applications within the IEEE Symposium Series on Computational Intelligence (IEEE SSCI), in 2013. His research interests are in the areas of cybersecurity, multimedia forensics, sensor security, steganography/steganalysis, information hiding, pattern recognition, and information fusion is funded by the National Science Foundation, National Academies, Ford Motor Company, Marelli USA, and other agencies. He has served as the Vice Chair of IEEE SEM, Chapter 16: Computational Intelligence from 2011 to 2017. He is also serving on several technical program committees, including the IEEE AVSS, Fake Multimedia, ICME, ICIP, MINES, ISPA, CCNC, ICASSP, and ICC. He is serving as an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and an Associate Editor for the springer *Journal on Signal Image and Video Processing*.

RIGEL GJOMEMO received the B.S. degree in information engineering from the Politecnico di Torino, and the Ph.D. degree in computer science from the University of Illinois at Chicago. He is currently a Research Associate Professor with the University of Illinois at Chicago. His research interests include intrusion detection, software analysis, Web security and privacy, and cybersecurity education.

V. N. VENKATAKRISHNAN is currently a Professor of computer science with the University of Illinois at Chicago. His research encompasses several aspects of software systems security for web, mobile and desktop platforms. His specific interests include vulnerability analysis, attack prevention and intrusion detection, and forensics and response.



RASHID ANSARI (Life Fellow, IEEE) received the B.Tech. and M.Tech. degrees in electrical engineering from the Indian Institute of Technology, Kanpur, India, and the Ph.D. degree in electrical engineering and computer science from Princeton University, Princeton, NJ, USA, in 1981. He is currently a Professor with the Department of Electrical and Computer Engineering, University of Illinois at Chicago. His research interests are in the general areas of signal processing and communications, with recent focus on image and video analysis, multimedia communication, and machine learning for healthcare and biomedical applications. Dr. Ansari served on the organizing and program committees of several past IEEE conferences and the Visual Communication and Image Processing (VCIP) Conferences. He was the General Chair of a VCIP Conference. He has served in editorial roles for several professional society journals, including as an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, and the IEEE SIGNAL PROCESSING LETTERS. He was on the Editorial Board of the *Journal of Visual Communication and Image Representation*.

AUN IRTAZA received the Ph.D. degree from FAST-NUCES, Islamabad, Pakistan, in 2016. During his Ph.D., he remained working as a research scientist in the Gwangju Institute of Science and Technology (GIST), South Korea. He became an Associate Professor, in 2017, and the Chair of the Department of Computer Science, University of Engineering and Technology (UET) Taxila, Pakistan, in 2018. He is currently working as a Visiting Associate Professor with the University of Michigan–Dearborn. He has more than 40 publications in IEEE, Springer, and Elsevier Journals. His research areas include computer vision, multimedia forensics, audio-signal processing, medical image processing, and Big data analytics.